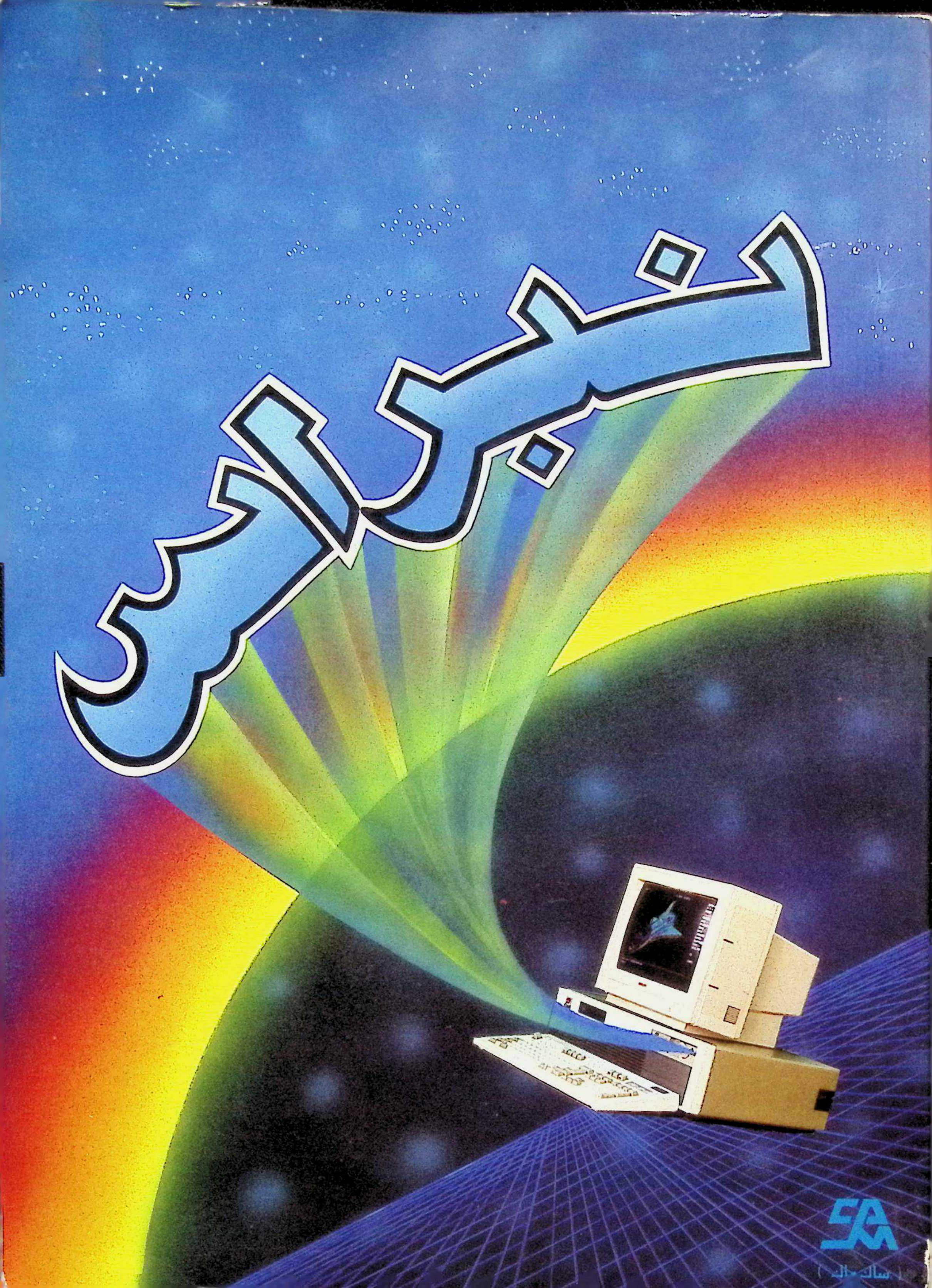

نبراس

المرجع الأساسي :

نبراس

(الوسيلة الحديثة للبرمجة باللغة العربية)

تأليف

المهندس/ سعد مقرن المقرن — المهندس / صالح منصور العساف

منفذ البرنامج

المهندس / صالح عبد الله الرميح

اصدار الشركة السعودية للحاسب الآلي (ساك ماك)

مطبوعات
الشركة السعودية للحاسب الآلي (ساك ماك)
ص.ب : ١٠٣٠٣ الرياض ١١٤٣٣
ت : ٤٦٤٩٥٣٥
فاكس : ٤٦٣١٨٣١
تلكس : ٤٠١٤٨٩ ساك ماك
الرياض : المملكة العربية السعودية

بسم الله الرحمن الرحيم

المحتـويات

oooooooo

بسم الله الرحمن الرحيم

مقـــدمـــة

لقد بدأ الحاسب الآلي يغزو جميع المجتمعات ومنها مجتمعنا العربي، مما جعل تعريب هذا العلم أمراً ضرورياً لمواكبة المجتمعات الأخرى ومنافستها.

والبرمجة باللغة العربية نادرة الوجود، لذلك فكرنا بعد سنوات من البحث والعمل في إيجاد لغة حديثة لتعلم البرمجة دون اللجوء الى اللغات الأجنبية ويعتبر هذا الكتاب المرجع الاساسي للغة «نبراس» وقد حاولنا جعله شاملا لجميع الأغراض العلمية والثقافية والادارية وقد راعينا الاختصار والوضوح.

ولغة «نبراس» مشابهة للغة الباسكال من حيث تركيبها وتأخذ معظم الخصائص والمميزات المتوفرة فيها وقد استعملنا فيها المصطلحات الشائعة في القواميس العلمية بحيث تكون مناسبة لاغراضها.

ولتبسيط هذه اللغة يجب معرفة طريقة استعمال البرنامج وأوامر التشغيل وأستعمال المحرر، وليس من الضروري تعلمها جميعا مرة واحدة ولكن بالممارسة سوف تعرف تلقائيا، بعد ذلك يتم التعرف على مكونات اللغة وتركيبها وطريقة بناء هيكل البرنامج وتركيب الجمل، وقد يواجه البعض صعوبات في البداية وهذا شيء طبيعي للمبتدئين، ومن الأفضل تصميم البرامج ومحاولة تطبيقها عمليا مما يعطي للمستخدم القدرة على رؤية الأخطاء وامكانية تصحيحها أو تجنبها أولاً بأول.

وهذا الجهد المتواضع يمثل مساهمة منا لحل مشكلة التعريب، وكلنا امل ان تنال رضاء كافة المبرمجين في العالم العربي..

والله ولي التوفيق

OOOOOOO

الباب الأول

مقدمة عن الحاسب الآلي INTRODUCTION TO COMPUTER

قبل أن نبدأ في توضيح لغات الحاسب الآلي، نود أن نوضح بعض الأشياء البسيطة عن مفهوم الحاسب الآلي، من حيث مكوناته وكيف تتفاعل هذه المكونات مع بعضها البعض، وسوف يكون حديثنا مقتصرا ففي هذا الباب سوف نتحدث عن المواضيع الآتية:

(١) أجزاء الحاسب الآلي.

(٢) البيانات الرقمية.

(٣) البرامج

(٤) الرموز.

١ــ١ اجزاء الحاسب الآلي

COMPUTER HARDWARE

يتكون الحاسب الآلي من عدة اجزاء ووحدات رئيسية وهي:

ــ وحدة المعالجة المركزية **(CPU)**.

ــ الذاكرة، ويوجد نوعان : ذاكرة عشوائية **(RAM)**، وذاكرة القراءة فقط **(ROM)**.

ــ وحدات التخزين كالاسطوانات الصلبة **(HARD DISK)**، والاسطوانات المرنة **(FLOPPY DISK)**.

ــ وحدات الادخال مثل لوحة المفاتيح **(KEY BOARD)**.

- وحدات الاخراج مثل الشاشة **(SCREEN)** وآلة الطباعة **(PRINTER)**.

وقد يحتوي الحاسب الآلي على وحدات إضافية أخرى، ولكي يؤدي الحاسب الآلي وظيفة كاملة في البرمجة فلا بد من توفر وحدة معالجة مركزية وذاكرة ووحدة ادخال ووحدة اخراج.

١ـ١ـ١ وحدة المعالجة المركزية

CENTRAL PROCESSING UNIT

تعتبر هذه الوحدة هي الدماغ المحرك للحاسب الآلي، ولكنها ليست ذكية كما نتصور، إلا أنها تستطيع تنفيذ واجراء بعض العمليات والأوامر البسيطة، فهي مثلاً تستطيع احضار رقمين مخزنين في مكان ما في الذاكرة وجمعهما مع بعضهما ثم وضع النتيجة في مكان آخر في الذاكرة.

تتكون وحدة المعالجة المركزية أيضا من أجزاء، وهذه الأجزاء تتصل ببعضها البعض عن طريق أسلاك كهربائية تعمل كقنوات **(BUSES)** تقوم بنقل التعليمات والبيانات، وهي لا تؤدي أي عمل من ذاتها، وإنما طبقا لبرنامج مخزن في مكان ما على الذاكرة الرئيسية والذي يصدر أوامره لوحدة المعالجة المركزية لتقوم بهذه المهمات.

١ـ١ـ٢ الذاكرة

MEMORY

تتكون الذاكرة الداخلية في الحاسب الآلي من دوائر كهربائية **(IC)** مدمجة ومصفوفة على خلية سيلكونية **(CHIP)**.

هناك نوعان من الذاكرة في الحاسب الآلي : النوع الأول هو **الذاكرة**

العشوائية (RAM) أو الرئيسية التي تخزن فيها البرامج والبيانات عندما يكون الجهاز في وضع التشغيل، وهذه الذاكرة يمكن تغيير محتواها، وتسمى عشوائية لأن الحاسب الآلي يستطيع القراءة والكتابة من أي موقع فيها بطريقة مباشرة بدون البحث في جميع محتويات الذاكرة بالتدريج للوصول إلى العنوان المطلوب، ويتم ذلك بواسطة وحدة المعالجة المركزية. والنوع الثاني هو **ذاكرة القراءة فقط (ROM)**، ويخزن فيها الأوامر وبعض البرامج التي تبنى داخل الحاسب الآلي، حيث ان وحدة المعالجة المركزية تستطيع قراءة هذه البرامج والأوامر المخزنة فيها فقط ولا يمكن أن تغير هذه الذاكرة أو الكتابة فيها.

١ــ١ــ٣ وحدات التخزين

MASS STORAGE

يحتوي الحاسب الآلي على سعة قصوى للذاكرة الرئيسية، ولهذا لا يمكن تخزين جميع البرامج والبيانات فيها، كما إنها لا تحتفظ بهذه المعلومات بعد إطفاء الجهاز حيث أن الحاسب الآلي يمسح جميع البيانات الموجودة في الذاكرة الرئيسية، ولتفادي هذه المشكلة تم إيجاد وحدات تخزين بديلة تحتفظ بالبرامج حتى بعد اطفاء الجهاز حيث تتسع هذه الوحدات لكمية كبيرة من المعلومات ويمكن نقلها وتداولها من حاسب لآخر ومن هذه الوحدات المتداولة ما يسمى باسطوانة التخزين حيث هناك نوعان منها : الأول، وهو الاسطوانة المرنة **(FLOPPY DISK)** وتعمل بواسطة محرك يحمل رأسا مغناطيسيا يؤدي وظيفة الكتابة والقراءة من الاسطوانة، ويمكن أن يحتوي الجهاز على أكثر من محرك واحد، وتتميز الاسطوانة بقدرتها على تخزين كمية كبيرة من البيانات، ويمكن تغييرها بأخرى عند نفاذ سعتها. والنوع الآخر هو الاسطوانة الصلبة **(HARD DISK)** ويمكن إضافتها في الجهاز مع محرك الاسطوانة المرنة

وتتميز بأنها أسرع من محرك الاسطوانة المرنة وتتسع إلى كمية هائلة من البيانات التي تفوقها بعشرات المرات، كما إنها توفر للمستخدم جميع البيانات التي يحتاجها بدلا من البحث عنها بين الأسطوانات المرنة.

١ـ١ـ٤ وحدات الادخال

INPUT DEVICES

وحدة الإدخال الرئيسية في الحاسب الآلي هي لوحة المفاتيح، ويمكن ادخال البيانات بوحدات أخرى كالفأر (MOUSE) واللوحة الرقمية (DIGITIZING TABLET) والموديم وشاشة اللمس، ويمكن أن نعرف وحدة الادخال على أنها وسيلة لادخال البيانات في الحاسب الآلي.

١ـ١ـ٥ وحدات الاخراج

OUTPUT DEVICES

وحدة الاخراج الرئيسية في الحاسب الآلي هي الشاشة. فعندما ندخل البيانات نستطيع أن نراها على الشاشة، ولكن البيانات لا تذهب إلى الشاشة مباشرة وإنما تذهب الى وحدة المعالجة المركزية حيث يتم تقييمها ثم ارسالها للشاشة. قد يحتوي الحاسب الآلي على وحدات اخراج أخرى كالآلة الطابعة والموديم، وتعرف وحدة الاخراج إنها وسيلة لاخراج البيانات من الحاسب الآلي.

١ـ٢ البيانات الرقمية

DIGITAL DATA

لنتخيل أن الحاسب الآلي يتكون من كمية كبيرة من مفاتيح كمفاتيح الاشعال (SWITCES) وكل مفتاح يحتوي على وضعين إما «شغال» أو «غير شغال».

عندما يكون المفتاح في وضع «شغال» فذلك يعني أن التيار الكهربائي يسري من خلال السلك إلى جزء آخر من الجهاز، وعندما يكون في وضع «غير شغال» فإن التيار لن يسري عبر هذا المفتاح ولنسمي الوضع «شغال» الرقم «١» والوضع «غير شغال» الرقم «٠».

المفتاح الواحد يستطيع أن يتحكم في خانة تخزين واحدة تسمى البت (BIT)، وبوضع مفتاحين بجانب بعضهما نحصل على أربعة احتمالات يمكن أن نجعلها تمثل أرقاما وهي:

مفتاح١	مفتاح٢	احتمال
غير شغال	غير شغال	صفر
شغال	غير شغال	١
غير شغال	شغال	٢
شغال	شغال	٣

وبوضع ثمانية خانات بجانب بعضها يمكن أن نحصل على خلية أو وحدة التخزين المسماة (بايت).. والتي تستطيع أن تعطي ٢٥٦ نتيجة أو احتمال. النظام الذي يستعمل هذه الطريقة لتمثيل البيانات يسمى **النظام الثنائي** (**BINARY SYSTEM**) حيث أساسه الرقم «**٢**»، ونظام الأرقام المعروفة لدينا هو النظام العشري أي أساسه الرقم «**١٠**»، حيث كل خانة في الرقم العشري تمثل عشرة أضعاف الخانة التي على يمينها.

ويمكن رؤية ذلك إذا فرضنا أن هناك عددا عشريا وآخر ثنائيا من ثمان أرقام فإن ترتيب خانات كل منها كالآتي:

عشري: ٠ ٩ ٨ ٣ ٥ ٤ ٦ ٥

$= (٠\times ١٠) + (٩\times ١٠^{١}) + (٨\times ١٠^{٢}) + (٣\times ١٠^{٣}) + (٥\times ١٠^{٤}) + (٤\times ١٠^{٥}) + (٦\times ١٠^{٦}) + (٥\times ١٠^{٧})$

ثنائي: ١ ١ ٠ ١ ٠ ٠ ١ ١

$= (١\times ٢^{٠}) + (١\times ٢^{١}) + (٠\times ٢^{٢}) + (١\times ٢^{٣}) + (٠\times ٢^{٤}) + (٠\times ٢^{٥}) + (١\times ٢^{٦}) + (١\times ٢^{٧})$

ويمكن التحويل من النظام العشري إلى النظام الثنائي والعكس.

مثال:

تحويل العدد ٦٥ من نظام عشري إلى نظام ثنائي:

٦٥ ÷ ٢ = ٣٢ + (١)

٣٢ ÷ ٢ = ١٦ + (٠)

١٦ ÷ ٢ = ٨ + (٠)

٨ ÷ ٢ = ٤ + (٠)

٤ ÷ ٢ = ١ + (٠)

٢ ÷ ٢ = ١ + (٠)

١ ÷ ٢ = ٠ + (١)

ويصبح العدد الثنائي = ١٠٠٠٠٠١

ومن هذا أصبح عدد الخانات الثمان عددا مناسبا لخلية الذاكرة، فإذا تخيلنا أن الذاكرة عبارة عن مصفوفة من ٦٥,٥٣٦ سطراً من المفاتيح وكل سطر يخزن ثمان خانات (بت) ليمثل رقما معينا أو رمزا يفهمه الحاسب الآلي. ويمكن القول أن الذاكرة تحتوي على ٦٥,٥٣٦ بايت، ونلاحظ أن في لغة الحاسب الآلي استخدام الكلمة «**كيلو**» وهي تساوي ١٠٢٤، ولذا تتكون الذاكرة من ٦٤ كيلوبايت.

تقوم وحدة المعالجة المركزية بأخذ الأوامر المخزنة في الذاكرة الرئيسية وذلك بطلب قراءة محتوى موقع معين بها حيث كل موقع يحمل عنوانا معينا يدل عليه في الذاكرة وتستطيع من خلاله قراءة وكتابة أي خانة في الذاكرة. وتمر هذه الأرقام من خلال الأسلاك والدوائر الكهربائية على هيئة صفر وواحد إلى وحدة المعالجة المركزية، وتتكون أوامر الوحدة من شفرة معينة تمكنها من نقل البيانات من مكان إلى مكان، واجراء عمليات جمع، وطرح، وضرب، وقسمة، وبعض من العمليات البسيطة الأخرى. وتسمى هذه الشفرة **بالشفرة الثنائية** (BINARY CODE) **أو لغة الآلة** (MACHINE LANGUAGE).

١ــ٣ برامج الحاسب الآلي

COMPUTER SOFTWARE

ترتبط البرامج ارتباطا وثيقا بوحدات الحاسب الآلي السابقة فكل منها لا يستغني عن الآخر، فالحاسب الآلي عديم الفائدة بدونها كما إنها عديمة الفائدة إذا لم يتوفر لدينا بعض وحدات الحاسب الآلي الرئيسية، ونقصد بهذه البرامج.. أي مجموعة الأوامر التي يقرأها الحاسب الآلي وتجعله يؤدي بعض الوظائف الرئيسية. ويحتفظ الحاسب الآلي ببرامج داخلية في الذاكرة الرئيسية وذاكرة القراءة فقط.. حيث إنها تصمم في مصنع الحاسب الآلي.

وتنقسم البرامج إلى فئتين:

(١) أنظمة التشغيل

(٢) البرامج التطبيقية

١ـ٣ـ١ أنظمة التشغيل

OPERATING SYSTEMS

إن المستخدم العادي لا يريد أن يعطي تعليمات للحاسب الآلي في كيفية قبول رمز من لوحة المفاتيح أو اظهار رمز على الشاشة أو ارسال رموز إلى آلة الطباعة بصياغة صحيحة أو كتابة بايت لاسطوانة التخزين، في هذه الحالة لن يملك المستخدم الوقت الكافي لحل مسائله الخاصة، ولهذا صممت أنظمة التشغيل لتسهل عليه أداء هذه العمليات بدون مشقة أو عناء، وبالإضافة إلى ذلك فهي تنظم الملفات والبيانات فتستطيع أن تنسخ او تلغي أو تنفذ ملفا كما يمكن تغيير اسم الملف، واصدار دليل للملفات كما تستطيع أيضا معرفة الوقت والتاريخ. ويعتمد نوع نظام التشغيل على نوع وحدة التحكم المركزية المتوفرة في الحاسب الآلي فمثلا إذا كان حاسبك الشخصي من نوع «**IBM XT**» أو متوافق معه فإنه يحتوي على وحدة التحكم (٨٠٨٨) ويمكن استخدام نظام التشغيل «**MS-DOS**»، أو «**PC-DOS**»، أو «**CP/M-86**» فيه.

١ـ٣ـ٢ البرامج التطبيقية

APPLICATION SOFTWARE

تتكون البرامج التطبيقية عامة من برامج تخدم اهدافا معينة، حيث بعضها تأتي مع الحاسب الآلي وبعضها يمكن شراؤه من الأماكن المخصصة. كما يمكن تصميم البرامج بواسطة الشخص نفسه أو عن طريق مبرمج أو شركة معينة، ومن البرامج التطبيقية المعروفة برامج معالجة الكلمات **(WORD PROCESSORS)** وبرامج قاعدة البيانات **(DATA BASE)** وأخرى، ونظرا للمشقة التي تواجه المبرمج في استخدام لغة الآلة لتصميمها فقد صمم لها لغة مختصرة

تتكون من أوامر مفهومة، وكل أمر يمثل عملية من عمليات الحاسب الآلي الرئيسية كالعمليات الحسابية، ونقل محتويات ذاكرة من مكان إلى آخر وتسمى اللغة المختصرة «**لغة التجميع**» (ASSEMBLY LANGUAGE) ولكل نوع من وحدات المعالجة المركزية لغة تجميع مستقلة ولكن جميعها متشابهة.. فإذا عرف المبرمج واحدة منها فسوف يعرفها جميعا.

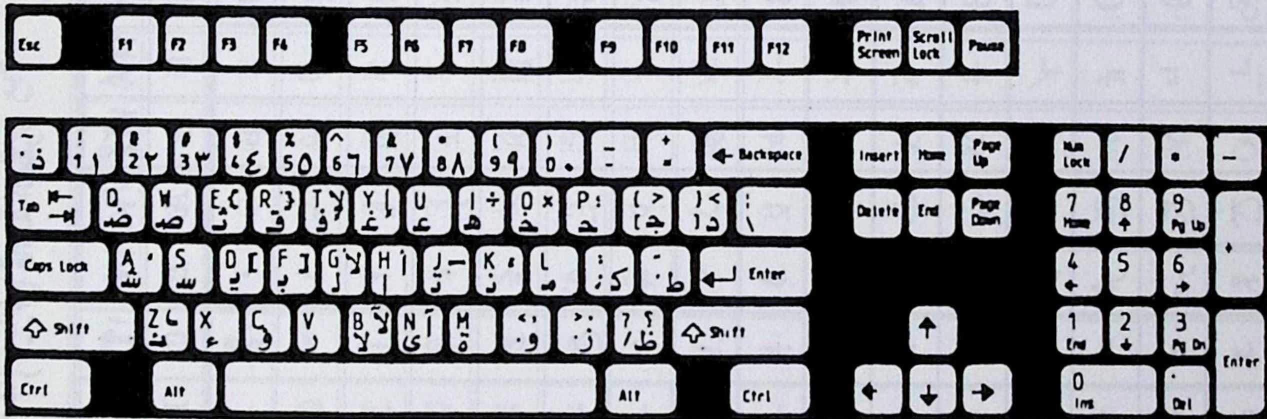
صممت بعض اللغات التي تسمى **باللغات العالية** (HIGH LEVEL LANGUAGE)، حيث أن السطر الواحد منها قد يفعل ما تفعله عدة سطور من لغة التجميع لتتناسب مع الاستعمال اليومي للمبرمج ولتوفر بعضا من الوقت عليه في كتابة العديد من الأسطر والأوامر، كما تحذف العديد من التفاصيل الفنية الدقيقة التي تحصل داخل الآلة ولا يحتاج المبرمج لمعرفتها، وهذه اللغات متوافقة مع معظم أجهزة الحاسب الآلي الشخصي بغض النظر عن نوع وحدة التحكم المركزية، ومن اللغات العالية الشائعة باللغة الانجليزية مثلا لغة الباسكال والبيسك والكوبول، أما في اللغة العربية فلا يوجد مع الاسف الا عدد محدود من اللغات وكل لغة مخصصة لنوع معين من أجهزة غير متوافقة مع أجهزة الحاسب الآلي الشخصي، فلغة صخر بيسك مثلا لا تعمل إلا على أجهزة صخر، ولغة الخوارزمي التي لا تعمل إلا على جهاز الرائد.

١ــ٤ الرموز

CHARACTERS

عندما يضغط المستخدم على مفتاح في لوحة المفاتيح، فالدوائر الكهربائية في وحدات الادخال تحول الرموز المدخلة إلى شفرات ثنائية وإرسالها إلى وحدة المعالجة المركزية، حيث تحول هذه الشفرات إلى رموز مفهومة ثم

ترسلها إلى وحدات الاخراج، وتتكون الرموز من حروف وأرقام ورموز خاصة وفي بعض الأحيان يوجد رموز خاصة للرسم، كما يوجد بعض الرموز التي تستخدم كإشارات لوحدات الحاسب الآلي وتسمى رموز التحكم، وكل رمز مخصص له شفرة ثنائية خاصة، فإذا خصصت مصانع الحاسب الآلي شفرات مختلفة لكل منها... فإن ذلك سيبب اختلاط في البيانات بين كل حاسب وآخر من الحاسبات الشخصية، ولحسن الحظ أن هذه المصانع اتفقت على شفرة الآسكي (ASCII) وتمثل هذه الشفرة ١٢٨ رمزا للحروف الإنجليزية والأرقام والرموز الاخرى المستخدمة بكثرة، وعند بداية استخدام اللغة العربية في الحاسب الشخصي قامت العديد من الشركات بوضع شفرات مختلفة عن الأخرى ولذلك اصدرت الهيئة العامة للمواصفات والمقاييس في المملكة العربية السعودية قائمة بالشفرة العربية القياسية التي يجب العمل بها من قبل الشركات والمصانع وتشمل أيضا ١٢٨ رمزا للحروف العربية والأرقام وبعض الرموز الخاصة.



شكل (١ــ١) لوحة المفاتيح

(١ــ٢) قائمة الشفرة

Dec	→	١٢٨	١٤٤	١٦٠	١٧٦	١٩٢	٢٠٨	٢٢٤	٢٤٠
↓	Hex	8	9	A	B	C	D	E	F
٠	0	°	β		٠	[illegible]	ذ	ـ	ـً
١	1	·	∞	─	١	ء	ر	ﻓ	ّ
٢	2	•	∅	ﺂ	٢	آ	ز	ﻗ	ﻧ
٣	3	√	±	ﻹ	٣	أ	ﺳ	ﻛ	ه
٤	4	▒	[illegible]	∝	٤	ؤ	ﺷ	ﻟ	ﻬ
٥	5	─	[illegible]	ﺄ	٥	ع	ﺻ	ﻣ	ى
٦	6	│	[illegible]	ﺈ	٦	ﺋ	ﺿ	ﻧ	ي
٧	7	┼	[illegible]	ﺎ	٧	ا	ط	ﻫ	ﻐ
٨	8	┤	[illegible]	ﺎ	٨	ﺑ	ظ	و	ق
٩	9	┬	ﻷ	ب	٩	ة	ﻋ	ى	ﻵ
١٠	A	├	ﻸ	ت	ف	ﺗ	ﻏ	ﻳ	ﻶ
١١	B	┴	ﻹ	ث	؛	ﺛ	!	ﺾ	ل
١٢	C	┐	ﻺ	،	ﺲ	ﺟ	┐	ﻌ	ك
١٣	D	┌	ﻻ	ج	ﺶ	ﺣ	÷	ﻐ	ي
١٤	E	└	ﻼ	ح	ﺺ	ﺧ	×	غ	■
١٥	F	┘	،	خ	؟	د	ع	ﻡ	

تابع شكل (١ـ٢) قائمة الشفرة

Dec	→	٠	١٦	٣٢	٤٨	٦٤	٨٠	٩٦	١١٢
↓	Hex	0	1	2	3	4	5	6	7
٠	0		►		0	@	P	~	p
١	1	☺	◄	!	1	A	Q	a	q
٢	2	♪	↕	"	2	B	R	b	r
٣	3	♫	‼	#	3	C	S	c	s
٤	4	☼	¶	$	4	D	T	d	t
٥	5	═	§	%	5	E	U	e	u
٦	6	║	▬	&	6	F	V	f	v
٧	7	╬	↨	'	7	G	W	g	w
٨	8	╣	↑	(	8	H	X	h	x
٩	9	╦	↓	)	9	I	Y	i	y
١٠	A	╠	→	*	:	J	Z	j	z
١١	B	╩	←	+	;	K	[	k	{
١٢	C	╗	∟	,	<	L	\	l	|
١٣	D	╔	↔	−	=	M	]	m	}
١٤	E	╚	▲	.	>	N	^	n	~
١٥	F	╝	▼	/	?	O	_	o	⌂

الباب الثاني

نظام تشغيل البرنامج
OPERATING THE PROGRAM

تمهيد

إن لغة نبراس العربية مصممة لتعمل على جميع أجهزة الحاسبات الشخصية «IBM» والمتوافقة معها ولهذا فمجالها أوسع من اللغات العربية الأخرى كما أنها لغة من لغات البرمجة العالية، وقد صمم لها معالج كلمات داخلي ومصغر حيث يحتوي على الكثير من المميزات التي تسهل على المبرمج تصميم البرامج، ومن الأفضل للمستخدم تعلم بعض الأوامر الرئيسية فيها وليس ضروريا أن يتعلمها جميعا مرة واحدة ولكن مع مضي الوقت واستمرار الممارسة يمكن معرفتها تلقائيا، ويجب استخدام أحد النظامين «MS-DOS 3.X» أو «PC-DOS 3.X» لتشغيل برنامج لغة نبراس، وسنفترض أن المستخدم يعرف قليلاً عن نظام تشغيل الجهاز. ففي هذا الباب سنوضح طريقة تشغيل البرنامج والمميزات المتوفرة فيه.

٢ــ١ طريقة التشغيل

RUNNING THE PROGRAM

قبل تشغيل برنامج نبراس يجب أن يكون نظام «MS-DOS 3.X» أو نظام «PC-DOS 3.X» مدارا في الجهاز لكي يتقبل أوامر تشغيل البرنامج، ولتشغيله.. يجب اتباع الخطوات التالية:

١) ضع اسطوانة برنامج نبراس داخل فتحة الاسطوانة، وذلك بعد تحديد مسمى المحرك وهو غالبا إما المسمى «:A» أو «:B».

٢) أكتب الأمر «NIBRAS»، ثم أضغط المفتاح «ENTER» لبدء تشغيل البرنامج ونلاحظ ظهور مقدمة البرنامج.

٣) تظهر بعد ذلك لائحة الاختيارات المخصصة للبرنامج، ونلاحظ أن كل اختيار يشمل على حرف مضيء، وهو الحرف اللازم ضغطه لتنفيذ الأمر.

```
محرك الأسطوانة : A          عربي اضغط F10للانجليزي
الدليل المشتغل : \
ملف العمل:
ملف رئيسي:
حرر    نفذ    ترجم ملف    الغاء ملف
خزن    دليل    طباعة ملف    انهاء
نص :بايت
حرة : ٦١٩٣٠ بايت
```

شكل ١ــ٣ لائحة الاختيارات

٢ـ١ـ١ استخدام الأوامر

USING COMMANDS

أن جميع الأوامر التي تتوفر في لائحة الاختيارات تخدم أهدافا معينة تساعد المبرمج وتسهل عليه الكثير من الأمور. وهذه الأوامر تعمل بضغط الحرف المضيء الذي يمثل كل أمر وبدون الحاجة إلى ضغط مفتاح العودة «RETURN»، وعند ضغط هذا الحرف ينفذ الأمر في الحال.

محرك الاسطوانة

ينفذ أمر محرك الاسطوانة بضغط الحرف «م» ويستخدم هذا الأمر لتغيير اسم محرك الاسطوانة، حيث يستطيع المبرمج ادخال أي اسم من «A» إلى «P» يليه مفتاح العودة <RETURN>، وغالبا ما يحتوي الحاسب الآلي الشخصي على أحد المحركين (A ، B) أو كلاهما معا، وإذا أراد المبرمج الأستمرار في نفس المحرك الحالي.. فاستخدام مفتاح العودة فقط سوف يبقيه في نفس المحرك. ونلاحظ أن المحرك الجديد لن يظهر على الشاشة، ولكن بضغط مفتاح الفراغ مثلا<SPACE> ستجعله يتغير، ويظهر على اللائحة. وفائدة هذا الأمر أن المبرمج يستطيع تخزين وقراءة البرامج والملفات على الأسطوانات الموجودة في أي محرك متوفر في الحاسب الآلي الشخصي.

أعمل الخطوات التالية:

- اضغط الحرف «م».
- اضغط المفتاح <F10>

- اكتب مثلا الحرف «C» أو أي حرف يطلق على محرك اسطوانة آخر في حاسبك الآلي.
- اضغط مفتاح العودة <RETURN>
- اضغط مفتاح الفراغ.

عربي اضغط F10 للانجليزي

تعني هذه الجملة أن مفاتيح الادخال في لوحة المفاتيح تعمل باللغة العربية، ويمكن تحويلها إلى اللغة الانجليزية باستخدام المفتاح الوظيفي <F10>، وهذه الميزة مفيدة لكتابة أسماء البرامج والملفات باللغة العربية، كما إنها متوفرة في معالج الكالمات (المحرر) أيضا.

ENGLISH اضغط F10 للعربي

هذه الجملة، هي نقيضة الجملة السابقة، فتعني أن مفاتيح الادخال في لوحة المفاتيح تعمل باللغة الانجليزية، ويمكن تحويلها إلى اللغة العربية باستخدام المفتاح الوظيفي <F10>، وهذه الميزة مفيدة لكتابة أسماء البرامج باللغة الإنجليزية، وكذلك لتغيير محرك الاسطوانة. وهي كما ذكرنا سابقا متوفرة في المحرر لتمكن المبرمج من ادخال الرموز الإنجليزية، ومن الأفضل عدم استعمال هذه الرموز في المحرر إلا عند الضرورة.

الدليل المشتغل

هذه الجملة توضح الدليل المستخدم الحالي، ويمكن تغييره بضغط الحرف «ش»، يقوم بعد ذلك بالسؤال عن اسم الدليل المراد احضاره وجعله الدليل المشتغل، فإذا كان غير موجود على اسطوانة التخزين، تقوم السماعة باصدار صوت يدل على عدم وجوده.

ملف العمل

ينفذ هذا الأمر بضغط الحرف «ع»، ويعمل على احضار الملف المراد العمل به حيث يمكن تحريره أو تنفيذه أو تخزينه، فإذا كان الملف غير موجود على اسطوانة التخزين.. فإن نظام تشغيل اللغة سوف يعتبر هذا الملف ملفا جديدا، ويجب أن لا يتعدى اسمه عن ثمانية رموز، وقد يحتوي اسم الملف على **نهاية تصنيفية** لا تتعدى ثلاثة رموز تلحق اسمه ويفصل بينهما النقطة «.» بحيث توضح هذه النهاية صنف الملف، وإذا لم توضع النهاية التصنيفية والنقطة فإن نظام تشغيل لغة نبراس سوف يضع النهاية النظامية «**.عرب**»، كما يمكن كتابة اسم الملف بدون نهاية تصنيف وذلك بكتابة اسمه ثم وضع نقطة بعده فقط، ويظهر اسم الملف في القائمة بعد ضغط مفتاح الفراغ <SPACE>

أمثلة:

برنامج	يصبح الاسم في الدليل «برنامج.عرب».
برنامج.	يصبح الاسم في الدليل «برنامج».
برنامج.نص	يصبح الاسم في الدليل «برنامج.نص».

يجب تجنب النهايات التصنيفية «COM»، و«BAK»، و«EXE» حيث أن نظام نبراس يستخدم هذه الأسماء لبعض الأغراض الخاصة التي سنعرفها فيما بعد. وعند الأنتهاء من تحرير ومعالجة الملف ورغبنا تغييره أو الخروج إلى نظام التشغيل الرئيسي «Dos» فان التحذير الآتي سيظهر على الشاشة لينبه المستخدم أنه لم يخزن الملف بعد:

((ملف غير مخزن هل تريد التخزين (ن/ل)؟.))

فإن ضغط الحرف «ن» فإن النظام سيخزن الملف على اسطوانة التخزين

المستعملة والموضحة في خانة «محرك الاسطوانة»، وإذا لم نرغب في تخزينه نضغط على الحرف «ل».

الملف الرئيسي

ينفذ هذا الأمر بضغط الحرف «**س**» ، ويستعمل لتعريف ملف رئيسي في نظام اللغة يختلف عن ملف العمل، فإذا كان الملف الرئيسي يحتوي على الموجهة {$م} .. حيث تستخدم هذه الموجهة فيه لربطه مع ملفات أخرى، ويمكن تحرير هذه الملفات واحدا بعد الآخر بدون تغيير الملف الرئيسي، فإذا نفذ الملف الرئيسي وكان ملف العمل يختلف عنه فان ملف العمل يخزن آليا، فإذا وجد خطأ في التنفيذ فإن الملف الذي حدث فيه الخطأ يصبح ملف العمل ليتمكن المستخدم من تحريره ومعالجته.

مثال:

برنامج رئيسي؛

{$م محتوى. صنف}

بداية

..

..

..

نهاية.

حرر

عند ضغط مفتاح الحرف «**ح**»، يعمل نظام نبراس على توفير معالج الكلمات

(المحرر) لتحرير الملف المخصص في خانة «ملف العمل» إذا كان موجودا وإلا فإنه سيطلب كتابة اسم الملف المراد تحريره، حيث يمكن تحرير ومعالجة الملفات وكتابة البرامج بواسطة هذا المحرر، ويحتوي هذا المحرر على الكثير من المميزات الخاصة والعمليات التي تتيح للمستخدم سهولة اجراء التحرير والمعالجة، وسنذكر هذه العمليات والمميزات بدقة في الفصل الخاص بالتحرير.

مثـال:

اكتب البرنامج التالي في المحرر:

برنامج تجربة؛

بداية

اكتب سطر ('هذا اول برنامج')،

نهاية.

بعد الإنتهاء اضغط على المفتاحين «Ctrl ئـ» للخروج من المحرر.

نفذ

وعند ضغط الحرف «**ن**»، يقوم نبراس بتشغيل برنامج يسمى «المترجم»، وهو البرنامج الذي يقوم بتحليل برامج هذه اللغة وتنفيذها فإذا لم يحدد ملف رئيسي فسوف ينفذ ملف العمل إذا كان موجودا وإن لم يجده فإنه سينفذ الملف الرئيسي وإن لم يجد شيئاً فإنه سيطلب كتابة اسم الملف المراد تنفيذه. ونعني بالكلمة «تنفيذ».. أي أن المترجم المبني في اللغة يقوم بترجمة البرنامج إلى اللغة التي يفهمها الحاسب الآلي ومن ثم يقوم الحاسب الآلي باجراء وتنفيذ العمليات المطلوبة منه في البرنامج ثم يقوم المترجم بترجمتها مرة أخرى من لغة الحاسب الآلي إلى لغة نستطيع أن نفهمها. ويمكن إيقاف عملية التنفيذ بالضغط على أي مفتاح.

ملف مترجم

عند الضغط على الحرف «ت»، يقوم النظام بعمل ملف «أمري» من ملف العمل ينتهي بالنهاية التصنيفية «COM.»، ويجب أن يكون ملف العمل خاليا من الأخطاء التنفيذية، ويتميز الملف التنفيذي أنه ينفذ في نظام التشغيل الرئيسي «Dos» وذلك بطباعة اسمه فقط.

ويفضل في هذه الحالة استعمال اسما للملف باللغة الانجليزية لأن عملية التعريب المستعملة سارية في نظام تشغيل نبراس فقط ولا تعمل في نظام التشغيل الرئيسي، كما أن البيانات الخارجة سوف تخرج بالحروف اللاتينية فإذا أراد المبرمج البيانات الخارجية أن تكون باللغة العربية فعلية أن يعرف في البرنامج الاجراء المختص بتحويل الحروف إلى العربية الموجود في الملف «حروف. ث» على أنه اجراء «خارجي» ويسمى هذا الاجراء «عربي» (راجع الباب المختص بالملفات (الخارجية) والذي يعمل على تغيير الحروف فإن المبرمج يريد إعادة الحروف اللاتينية للنظام الرئيسي ولذلك فعليه أن يعمل نفس الطريقة السابقة ولكن مع الاجراء الخارجي «انجليزي» ويضع جملة الاجراء في نهاية البرنامج.

خـزن

عند الضغط على الحرف «خ»، يعمل النظام على تخزين ملف العمل الحالي على اسطوانة التخزين الموجودة في محرك الاسطوانة المستخدمة، وإذا كان هناك نسخة قديمة من هذا الملف بنفس الاسم.. فإنها سوف تحول إلى نسخة إحتياطية وذلك باعطاء الملف نفس الاسم ولكن باسم تصنيفي «ص م».

الدليـل

عند الضغط على الحرف «ل»، يوفر النظام قائمة الدليل، ويمكن أن تحدد

اسم محرك الاسطوانة واسم الدليل المرغوب فيه مثل:

الدليل: ‏B:\١ *٠عرب

وتعني أن المراد قائمة ملفات الدليل الرئيسي على محرك الاسطوانة (B) والتي نهاياتها التصنيفية (عرب). وإذا ضغط مفتاح العودة فقط بعد اصدار أمر الدليل «ل» فان نبراس سوف يعطي قائمة الملفات الموجودة على الدليل الحالي في محرك الاسطوانة الحالية.

الغاء ملف

يعمل هذا الأمر بالضغط على الحرف «غ»، بحيث يهيىء للمستخدم سهولة التخلص من الملفات التي لا حاجة لها بدون الخروج من نظام اللغة وقد صمم هذا الامر لتسهيل عملية الغاء الملفات في النظام حيث أن نظام التشغيل الأساسي لا يتعامل مع اللغة العربية، وقد تكون غالبية أسماء الملفات المراد الغاؤها مكتوبة باللغة العربية حيث يصعب كتابتها في نظام التشغيل الأساسي.

طباعة ملف

عند ضغط الحرف «ط» يقوم بطباعة ملف العمل الموجود في الذاكرة الرئيسية أو يطلب اسم الملف المراد طبعه إذا لم يوجد ملف للعمل.

انهاء.

عند ضغط الحرف« أ »، يعمل الأمر على اعطاء اشارة الانتهاء من لغة نبراس والخروج منها، وإذا كان هناك ملف للعمل لم يحفظ بعد فسوف يعطي تحذيراً يبين أن الملف لم يحفظ بعد.

٢ ــ٢ التحرير والمعالجة

EDITING AND PROCESSING

بعد معرفة أوامر القائمة الرئيسية وكيفية استخدامها، نتعرف في هذا الجزء على معالج الكلمات الداخلي وكيفية استخدام أوامره وخواصه ــ وهو جزء من نظام لغة نبراس ــ فُبالامكان كتابة البرنامج وتحريره. وعمل أي خاصة من خواصه بدون الخروج من النظام مما يجعل العمل وكتابة البرنامج اسهل بكثير من استخدام معالج كلمات خارجي.

إن معالج الكلمات الداخلي (المحرر) مخصص ومعد لاستخدامه في كتابة البرامج العربية، يمكن استعماله كمعالج كلمات لتحرير النصوص لأنه يعطي جميع خصائص المعالج العربي فمثلاً:

عند اختيار أداة الأمر (ح) من القائمة الرئيسية تظهر الشاشة إما خالية (أي لا يوجد ملف) إذا كان الملف جديد تظهر عبارة «ملف جديد»، أو ظهور البرنامج المراد تحريره على الشاشة. وتظهر في أعلى الشاشة معلومات تبين حالة البرنامج «خط الحالة» كما هو موضح في الشكل الآتي:

س سطر ع عمود: اضف هامش : اسم الملف عربي F10 للانجليزي

شكل ٢ ــ١ خط الحالة لمعالج الكلمات الداخلي

وتعني كل من هذه البيانات الموجودة في خط الحالة «STATUS LINE» الآتي:

س سطر:

يوضح الحرف (س) رقم السطر الذي يقع عليه المؤشر، مبتدئا العد من

سطر رقم ١.

ع عامود:

يوضح الحرف (ع) رقم العمود الذي يقع عليه المؤشر ابتداء من أقصى اليمين.

أضف:

تدل هذه الكلمة على حالة إضافة الرموز في النص أو السطر، فالمحرر يضيف أي رمز من رموز لوحة المفاتيح في موضع المؤشر، وإذا كان هناك رموز محررة فإنها تتجه إلى يسار المؤشر مع باقي السطر، وهذه هي الحالة النظامية للمعالج، فهي تظهر على خط الحالة عند بداية التحرير، وبالامكان ابدالها إلى الحالة «كتابة فوقية» بالضغط على المفتاحين (Ctrl) (V) أو (Ctrl) (ر) والتي تكتب على النص وتلغي ما سبق كتابته من قبل، ويمكن الرجوع إلى الحالة «أضف» بضغط (Ctrl) ،(ر> أو ضغط <INS>.

هامش:

تدل هذه الحالة على الوزن الآلي للنص، فعند البدء من نقطة معينة من الشاشة والضغط على مفتاح الإعادة (RETURN) فان المؤشر يصبح في نفس العمود ولكن في السطر الذي يليه، وبالامكان تغيير هذه الحالة بالضغط على (Ctrl) (ض) (هـ).

ملف:

توضح هذه العبارة اسم المحرك المستعمل واسم الملف.

عربي اضغط F10 للانجليزي:

تبين نوع اللغة المستعملة للكتابة وطريقة الانتقال من اللغة العربية إلى اللغة

الانجليزية والعكس، فإذا كانت اللغة العربية مستخدمة فإن اللون الخلفي لكلمة «عربي» مختلف، والضغط على المفتاح F10 يغير الكتابة إلى اللغة الانجليزية وتتغير الجملة إلى «انجليزي للعربي» ويتغير اللون الخلفي لكلمة «انجليزي».

٢ ــ ٢ ــ ١ أوامر التحرير

EDITING COMMANDS

يهيء معالج الكلمات الداخلي للمبرمج قرابة خمسين أمرا ليجعل التحكم والانتقال من مكان داخل النص امراً سهلاً وكذلك نقل جزء من النص إلى مكان آخر، أو ملف آخر. وكل مجموعة يربطها علاقات داخلية خاصة مع بعضها سوف نتكلم عن كل منها وخواصها على حدة، وهذه الأوامر سهلة وبسيطة وأفضل طريقة لممارستها هو بدء تشغيل لغة نبراس، وتحرير نص جديد والقيام بالقراءة والتطبيق.

وتصنف أوامر معالج الكلمات الداخلي إلى أربعة أقسام وهي:

١ ــ أوامر حركة المؤشر.

٢ ــ أوامر الإضافة والحذف.

٣ ــ أوامر الكتلة النصية.

٤ ــ أوامر متنوعة.

وللخروج من المعالج يمكن الضغط على المفاتيح (Ctrl) (ن) (ي) كالآتي:

ــ اضغط المفتاح (Ctrl)

ــ اضغط المفتاح (ن) ويكون المفتاح (Ctrl). مضغوطاً أيضاً

ــ اضغط المفتاح (ي)

٢ ـ ٢ ـ ١ ـ ١ أوامر حركة المؤشر

CURSOR MOVEMENT COMMANDS

صممت مفاتيح تحريك المؤشر في نظام معالج الكلمات لتتواجد في جهة واحدة من لوحة المفاتيح المستعملة في حركة المؤشر.. نجد أن الحروف (ش س ي ب ث ق ء) أو (A S D F E R X C) جميعها متواجدة في جهة واحدة من اللوحة وكذلك توجد بالقرب من مفتاح (Ctrl)، وهو المفتاح الذي يضغط أولاً قبل ضغط أي مفتاح. وإذا نظرنا إلى اختيار المفاتيح نجد الحروف الآتية:

ث E

س D   ش S

ء X

قد وضعت بحيث أن النظر اليها يجعل التعرف على خاصية كل مفتاح أمر يسير، فمثلاً (Ctrl) (ث) يحرك المؤشر إلى الأعلى، (Ctrl) (ء) يحرك المؤشر إلى الأسفل، (Ctrl) (ش) إلى اليسار، (Ctrl) (س) إلى اليمين، وجميعها مصممة على هذا الأسلوب. كما يمكن استعمال مفاتيح الأسهم الموجودة في الجهة اليمنى من لوحة المفاتيح بدلا منها.

رمز إلى اليسار   **(Ctrl) (ش)**

لتحريك المؤشر رمزاً واحداً إلى اليسار. ويمكن أيضا استعمال المفتاح رقم (٤) في لوحة المفاتيح.

رمز إلى اليمين (Ctrl) (س)

لتحريك المؤشر رمزاً واحداً إلى اليمين، أو باستعمال المفتاح رقم (٦) في لوحة المفاتيح.

كلمة إلى اليسار (Ctrl) (ي)

لتحريك المؤشر إلى بداية الكلمة اليسرى. والكلمة عبارة عن سلسلة من الرموز منتهية بأحد الرموز («»‘;:’+-/$’ فراغ) وبالإمكان الحصول على نفس الخاصية بالضغط على (Ctrl)، (٤) .

كلمة إلى اليمين (Ctrl)، (ب)

لتحريك المؤشر إلى بداية الكلمة اليمنى. وبالامكان الحصول على نفس الخاصية بالضغط على (Ctrl) (٦) .

سطر إلى أعلى (Ctrl > (ث)

لتحريك المؤشر إلى السطر العلوي من الصفحة. فإذا كان المؤشر على السطر العلوي وفي صفحة غير الصفحة الأولى فإن الشاشة تتدرج سطر واحد إلى الأسفل.

سطر إلى الأسفل (Ctrl) (ء)

لتحريك المؤشر إلى السطر الأسفل، فإذا كان المؤشر على السطر ما قبل الأخير فإن الشاشة تتدرج سطر واحد إلى الأعلى.

تدرج إلى الأعلى (Ctrl) (ص)

لتدريج النص إلى الأعلى بمقدار سطر واحد باتجاه بداية النص. في هذه الحالة جميع أسطر النص تتدرج إلى الأسفل مع ثبوت المؤشر في مكانه.

تدرج إلى الأسفل (Ctrl) (لا)

لتدرج النص إلى الأسفل بمقدار سطر واحد. في هذه الحالة جميع النص يتدرج إلى الأعلى مع ثبوت المؤشر في مكانه.

صفحة إلى الأعلى (Ctrl) (ق)

لتحريك المؤشر صفحة واحدة إلى الأعلى مع الاحتفاظ)سطر الأخير وبالإمكان استعمال (Pgup) .

صفحة إلى الأسفل (Ctrl) (ؤ)

لتحريك المؤشر صفحة واحدة إلى الأسفل مع الاحتفاظ بالسطر الأخير. ويمكن استعمال< End Ctlr >.

بعد الانتهاء من حركة المؤشر المبدئية، والتي تتطلب الضغط على (Ctrl) مع حرف واحد من الحروف المعرفة للأوامر، أو باستعمال مفاتيح الأسهم المتواجدة في الجهة اليمنى، سوف نوضح أوامر انتقال المؤشر والتي تتطلب الضغط على مفتاحين مع مفتاح التحكم (Ctrl) لتنفيذ العملية، فقد تم اختيار الحرف (ض) مع (Ctrl) لكي يضغط أولا مع أحد الحروف الآتية:

(س ي ث ء ق ؤ) أو (S D E X R C).

فإذا نظرنا إلى موقع الحروف على لوحة المفاتيح نجدها كالآتي:

ق R ثE

ي D س S

ؤ C ء X

إلى نهاية السطر (Ctrl) (ض) (س)

لتحريك المؤشر إلى نهاية السطر، كما يمكن استعمال المفتاح الوظيفي (F2) لجعل العملية أسهل.

إلى بداية السطر (Ctrl) (ض) (ي)

لتحريك المؤشر إلى بداية السطر، كما يمكن استعمال المفتاح (F1).

إلى أعلى الشاشة (Ctrl) (ض) (ث)

للانتقال بالمؤشر إلى الجزء العلوي من الشاشة، كما بالإمكان استعمال المفتاح (Home) (CTRL).

إلى أسفل الشاشة (Ctrl) (ض)

للانتقال بالمؤشر إلى الجزء السفلي من الشاشة، كما يمكن استعمال المفتاح (End)(CTRL).

إلى أول الملف (Ctrl) (ض) (ق)

للانتقال إلى أول رمز من الملف كما يمكن استعمال المفتاح (Home).

إلى نهاية الملف (Ctrl) (ض) (ؤ)

للانتقال إلى آخر رمز من الملف، كما يمكن استعمال المفتاح (End)

إلى بداية الكتلة النصية (Ctrl) (ض) (لا)

للانتقال إلى بداية الكتلة النصية التي سبق تعليمها من قبل بالعملية (Ctrl) (ن) (لا)، وهي لا تتطلب اظهار الكتلة على الشاشة (راجع اخفاء واظهار الكتلة النصية).

إلى نهاية الكتلة النصية (Ctrl) (ض) (ن)

للانتقال إلى نهاية الكتلة النصية التي سبق تعليمها من قبل بالعملية (Ctrl) (ن) (ن)، وهي لا تتطلب اظهار الكتلة على الشاشة.

إلى آخر موضع للمؤشر (Ctrl) (ض) (ح)

لتحريك المؤشر إلى آخر موضع له، فمثلاً إذا كان موضعه السطر ٤ والعمود ٩ وأجريت عملية تنفيذ للبرنامج، فعند الرجوع إلى التحرير فإن المؤشر يصبح في أول النص، وباستعمال المفاتيح (Ctrl) (ض) (ح) يذهب المؤشر إلى السطر ٤ والعمود ٩.

٢ ــ ٢ ــ ١ ــ ٢ أوامر الاضافة والحذف

INSERT AND DELETE COMMANDS

هذه الأوامر مخصصة لحذف وإضافة رمز، أو كلمة أو سطر، أو كتلة نصية، كما يمكن إعادة السطر الذي حدث فيه التغيير إلى وضعه السابق ما دام المؤشر لا يزال في السطر الذي تمت فيه العملية، أما إذا استعمل أمر من الأوامر الأخرى فانه من المستحيل إعادة الوضع السابق للسطر، ولإعادة السطر إلى وضعه السابق يجب الضغط على المفاتيح (Ctrl) (ض) (م).

أضف/ كتابة فوقية (Ctrl) (ر)

لاستعمال عملية الإضافة وهي الحالة النظامية «DEFAULT» لمعالج الكلمات ــ أي إنها الحالة التي يفترضها البرنامج عند تشغيله ــ ولاستعمال الكتابة الفوقية يجب ضغط المفتاحين (Ctrl) (ر). وبالامكان استعمال المفتاح المتواجد على لوحة المفاتيح في الجهة اليمنى (Ins).

عند ظهور الكلمة «أضف» على خط الحالة فان أي رمز تتم طباعته يضاف إلى النص مع اتجاه النص إلى يسار المؤشر خانة واحدة. أما عند ظهور كلمة «كتابة فوقية» ففي هذه الحالة فإن أي رمز تتم طباعته سيكتب على الرمز الذي عليه المؤشر.

اضف سطر (Ctrl) (ى)

تستعمل لإضافة سطر إلى النص، سواء كانت حالة الكتابة فوقية أو إضافية.

حذف الرمز في موضع المؤشر (Del)

لحذف الرمز في موضع المؤشر، وجميع الرموز الموجودة على يسار المؤشر تنتقل خانة واحدة إلى اليمين.

حذف الرمز الأيمن (Back)

لحذف الرمز الأيمن من موضع المؤشر، وبعد هذه العملية جميع الرموز الموجودة على يسار المؤشر تنتقل خانة واحدة إلى يمينه، والمفتاح (Back) يكون غالبا المفتاح الأول من الصف الأول في لوحة المفاتيح.

حذف الرمز أعلى المؤشر (Ctrl) (ل)

لحذف الرمز الموجود أعلى المؤشر وانتقال الرموز على يسار المؤشر خانة واحدة إلى اليمين.

حذف سطر (Ctrl) (غ)

لحذف السطر المتواجد عليه المؤشر وانتقال جميع النص الواقع تحت المؤشر سطرا واحدا إلى الأعلى، ويصبح موقع المؤشر في أقصى اليمين.

حذف إلى آخر السطر (Ctrl) (ض) (غ)

لحذف جميع النص من موقع المؤشر إلى نهاية السطر.

٢ ــ ٢ ــ ١ ــ ٣ أوامر الكتلة النصية

BLOCK COMMANDS

الكتلة النصية عبارة عن مجموعة من الرموز قد تكون إما حرفا أو كلمة،

أو عدة أسطر، أو مجموعة صفحات وتعلم الكتلة النصية بوضع علامة على أول رمز بالكتلة لتدل على بداية الكتلة، ووضع علامة نهاية الكتلة على آخر رمز فيها، بعد ذلك يمكن اجراء العديد من العمليات كنقلها، وحذفها وأخفاؤها، ونسخها، وتحريكها، أو كتابتها إلى ملف في اسطوانة أخرى.

تعليم بداية الكتلة النصية (Ctrl) (ن) (لا)

لتعليم بداية الكتلة النصية، توضع علامة على اول رمز فيها وهذه العلامة غير مرئية، كما يجب وضع علامة نهاية الكتلة على آخر رمز فيها، ويمكن استعمال المفتاح الوظيفي (F3).

تعليم نهاية الكتلة النصية (Ctrl) (ن) (ن)

لتعليم نهاية الكتلة النصية توضع علامة في نهايتها على آخر رمز، وهذه العلامة غير مرئية، ولكن الكتلة المعلمة تصبح بلون خلفي آخر غير لون النص الخلفي، ويجب تعليم بداية الكتلة أولا قبل اجراء هذه العملية. كما يمكن استعمال المفتاح الوظيفي (F4).

تعليم كلمة (Ctrl) (ن) (ب)

الكلمة معرفة بأنها سلسلة من الرموز منتهية بأحد الرموز "()،؛: + - / $ فراغ، بعد تعليم الكلمة تعامل كما تعامل الكتلة النصية وتطبق جميع خصائصها عليها.

اخفاء واظهار الكتلة النصية (Ctrl) (ن) (ا)

هذه العملية تخفي وتظهر الكتلة إذا كانت معلمة، فبالإمكان اخفاؤها أو اظهارها على الشاشة، وعندما تكون الكتلة مخفية فإن خواص الكتلة (الحذف، النسخ، النقل، الكتابة إلى الاسطوانة) غير فعالة، أما خواص المؤشر (الانتقال

إلى بداية ونهاية الكتلة) فإنها فعالة في كلا الحالتين.

نسخ الكتلة النصية **(Ctrl) (ن) (و)**

يقوم بوضع نسخة من الكتلة التي سبق تعليمها في موقع المؤشر، والكتلة المعلمة تبقى كما هي بدون تغيير وتصبح الكتلة الجديدة هي الكتلة المعلمة، ويمكن استعمال المفتاح (F5).

نقل الكتلة النصية **(Ctrl) (ن) (ر)**

لنقل كتلة سبق تعليمها من مكانها الأول ووضعها في المكان الجديد في موقع المؤشر. فالكتلة المعلمة تختفي من مكانها السابق وتبقى الكتلة الجديدة هي المعلمة.

الغاء الكتلة النصية **(Ctrl) (ن) (غ)**

لحذف كتلة سبق تعليمها حذف نهائي، ولا يمكن اعادتها مرة أخرى.

قراءة الكتلة النصية من الاسطوانة **(Ctrl) (ن) (ق)**

لقراءة ملف من الاسطوانة، ووضعه في موضع المؤشر، ولا ينبغي أن يكون الملف معلم ككتلة، إذ إنه يعامل ككتلة واحدة، كما يجب اعطاء الاسم الكامل للملف عند طلب هذه الخاصية.

كتابة الكتلة النصية إلى الاسطوانة **(Ctrl) (ن) (ص)**

تستعمل هذه العملية لكتابة كتلة معلمة من قبل إلى الاسطوانة، بعد كتابة الكتلة إلى الاسطوانة تعامل كملف مستقل، ولذلك يجب اعطاء اسم للملف عند السؤال عنه. وإذا كان الملف موجودا على الاسطوانة ستظهر جملة تحذيرية تدل أن هناك ملف موجود بهذا الاسم.

٢ ــ ٢ ــ ١ ــ ٤ أوامر متنوعة

MISCELLANEOUS COMMANDS

يتوفر في المعالج أوامر متنوعة، لا تنتمي إلى أي من المجموعات السابقة، ولا تربطها علاقة معينة، كما هو الحال في المجموعات السابقة، ولكن هذه الأوامر مهمة جدا لجعل عملية التحرير عملية سهلة وهي:

انهاء التحرير **(Ctrl) (ن) (يـ)**

لانهاء التحرير والرجوع إلى القائمة الرئيسية مع بقاء الملف المحرر بالذاكرة الداخلية للحاسب الآلي بدون تخزين، وتتم عملية التخزين من القائمة الرئيسية.

التسطير **(Ctrl) (هـ)**

لا يوجد موضع معين للتسطير داخل المعالج، ولكنه يوزن آليا على بداية أول كلمة تقع على السطر الذي أعلى المؤشر، هذه الطريقة توفر للمبرمج تسطيرا آليا يجعل تنظيم البرنامج سهلا، كما يمكن استعمال المفتاح (TAB).

تشغيل والغاء التسطير الآلي **(Ctrl) (ض) (هـ)**

عند استعمال هذه العملية، تظهر كلمة «هامش» على خط الحالة، أما إذا لم تظهر هذه الكلمة، فان التسطير الآلي لاغي، وعند ضغط مفتاح الإعادة، فان المؤشر ينتقل إلى العمود الأول من السطر التالي، أما عند ظهور العبارة «هامش» فإن التسطير الآلي فعال، وعند ضغط مفتاح الإعادة، فإن المؤشر ينتقل إلى السطر التالي في نفس العمود الذي تقع عليه أول كلمة في السطر السابق ويمكن تغيير الوزن بتغيير موقع العمود الذي يبدأ منه السطر.

اعادة السطر **(Ctrl) (ض) (م)**

هذا الأمر يتيح للمبرمج فرصة لإعادة السطر الذي تم تغييره طالما المؤشر

يقع على نفس السطر الذي وقع فيه التغيير، وهذه العملية مهمة جدا، فعند تغيير السطر عن طريق الخطأ يمكن إعادته إلى وضعه السابق قبل التغيير وهذه العملية لا تعيد السطر المحذوف.

البحث **(Ctrl) (ض) (ب)**

تستعمل هذه الخاصية للبحث عن كلمة أو جملة لا تزيد على ٣٠ رمزا، ويختفي خط الحالة وتظهر جملة طلب ادخال الكلمة أو الجملة المراد البحث عنها، ويمكن للجملة ان تحتوي على حروف ورموز عربية أو لاتينية، كما بالامكان تحرير أو تعديل الجملة المراد البحث عنها باستعمال أوامر تحريك المؤشر.

بعد الانتهاء من ادخال الجملة المراد البحث عنها، يجب الضغط على مفتاح العودة (Enter) لتأتي جملة «الاختيارات» لتتيح للمستخدم عدة اختيارات هي:

«خ» ابحث إلى الخلف مبتدئا من موقع المؤشر إلى بداية النص.

«ر» ر= رقم صحيح، ابحث عن الكلمة التي حدوثها في النص يوافق هذا الرقم فقط.

«م» مساواة، وهذا الاختيار يستعمل بالنص اللاتيني، أي إن الحروف جميعها متساوية لا فرق بين الحروف الكبيرة أو الصغيرة فعند ادخال كلمة «HELLO» فإن المؤشر يقف عند كلمة «hello».

«ك» ابحث عن الكلمة المطابقة جميع حروفها للكلمة المطلوبة فقط، أي عند ادخال كلمة «اسلامية» فإن المؤشر لن يقف عند كلمة «اسلامي».

كما يمكن دمج هذه الاختيارات مع بعضها بشرط أن لا تسبب اي تناقض فيما بينها.

أمثلة:

خ م ابحث إلى الخلف مع عدم التفريق بين الحروف الكبيرة والصغيرة بالنص اللاتيني.

٢٠ ابحث عن الكلمة المطابقة رقم ٢٠.

١٠خ ابحث إلى الخلف من موقع المؤشر و متجها إلى أعلى النص فقط لعاشر مرة مطابقة للنص المراد البحث عنه.

وعند وقوف المؤشر عند الكلمة المطلوبة يجب الضغط على (Ctrl) (م) لإعادة البحث، كما يمكن إيقاف البحث باستعمال (Ctrl) (ع).

البحث والتغيير (Ctrl) (ض) (ش)

تستعمل هذه الخاصية للبحث عن جملة أو كلمة وتغييرها، بحيث لا تزيد عن ٣٠ رمزاً، ويختفي خط الحالة وتظهر جملة طلب ادخال الكلمة أو الجملة المراد تغييرها، ويمكن للكلمة أن تحتوي على حروف ورموز عربية أو لاتينية، كما بالإمكان تحرير أو تعديل الكلمة المراد البحث عنها باستعمال أوامر تحريك المؤشر، بعد الإنتهاء من ادخال الكلمة المراد تغييرها، يضغط على مفتاح العودة (Enter) لتأتي الجملة «يحل محله» ثم يتم ادخال الكلمة أو الجملة الجديدة وبالضغط على مفتاح العودة (Enter) تظهر كلمة «اختيارات» وهذه الاختيارات هي:

«خ» ابحث وغير جميع الكلمات الموجودة متجها من موقع المؤشر إلى أعلى النص.

«ج» ابحث وغير جميع الكلمات الموجودة داخل النص.

«ر» ر= رقم صحيح. ابحث وغير الكلمة بعدد هذا الرقم فقط.

«ب» ابحث وغير بدون السؤال الاختياري «يحل محله (ن/لا)».

«م» مساواة، وهذا الاختيار يستعمل بالنص اللاتيني، أي أن الحروف جميعها متساوية ولا فرق بين الحروف الكبيرة أو الصغيرة، كما ذكرنا سابقا.

«ك» ابحث وغير الكلمة المطابقة جميع حروفها للكلمة المطلوبة فقط، كما ذكرنا سابقا في عملية البحث.

أمثلة:

خ م   ابحث وغير إلى الخلف مع عدم التفريق بين الحروف الكبيرة والصغيرة بالنص اللاتيني.

٢٠   ابحث وغير أول ٢٠ كلمة تطابق إلى المطلوب.

١٠خ   ابحث وغير إلى الخلف مبتدئا من موقع المؤشر متجها إلى بداية النص فقط لأول ١٠ كلمات تطابق النص المطلوب.

ج ب   ابحث وغير جميع الكلمات الموجودة بالنص بدون السؤال.

عند وقوف المؤشر عند الكلمة المطلوبة يجب الضغط على (Ctrl) (م) لإعادة البحث والتغيير، كما يمكن توقيف العملية باستعمال (Ctrl) (ع).

إعادة البحث أو البحث والتغيير (Ctrl) (م)

لإعادة آخر عملية البحث أو البحث والتغيير، كما لو أن المعلومات قد ادخلت من جديد.

إلغاء الأمر (Ctrl)(ع)

لإلغاء الأوامر أو التوقف عن عملية كعملية البحث أو السؤال في عملية البحث والتغيير «يحل محله (ن/لا)»، أو في إحدى عمليات قراءة ملف كتابته في ملف آخر.

OOOOOOO

الباب الثالث

مكونات اللغة LANGUAGE ELEMENTS

تمهيد:

تتعامل برامج الحاسب الآلي مع العديد من البيانات، فالبرامج عبارة عن وسيلة للتفاهم مع الحاسب الآلي، وتعتبر لغة نبراس من اللغات العالية للتفاهم مع الحاسب الآلي وجعله يؤدي الأهداف التي ننشدها، ولكل لغة من لغات الحاسب الآلي أو حتى لغات الإنسان المتداولة يوجد لها عناصر ومكونات، فعناصر اللغة العربية مثلا هي الأحرف الهجائية والأرقام، فتدمج الأحرف الهجائية ببعضها لتكون كلمات مفهومة للإنسان وتدمج الكلمات ببعضها أيضا مستخدما قواعد اللغة لتكون جملا. تتكون كذلك لغة البرمجة هذه من انواع مختلفة من العناصر التي سنوضحها في هذا الباب وهي الرموز الأساسية، والكلِمات المحجوزة، والأسماء التعريفية، والأسماء التعريفية القياسية، والثوابت الرمزية، ورموز التحكم، والجمل التوضيحية.

٣ـ١ الرموز الأساسية

CHARACTERS

الرموز الأساسية في اللغة مقسمة إلى أحرف وأرقام، ورموز خاصة:

الأحرف

LETTERS

جميع الأحرف العربية واللاتينية الموجودة على لوحة المفاتيح والرمز «ـــ».

الأرقـام

NUMBERS

٩ ٨ ٧ ٦ ٥ ٤ ٣ ٢ ١ ٠

الرموز الخاصة

SPECIAL SYMBOLS

+ - . / = ^ > < ( ) [ ] { } ، : ؛ ' # $

قد تكون بعض هذه الرموز غير موضحة على لوحات المفاتيح ولذلك من الأفضل الرجوع إلى شكل رقم (٢) في الباب الأول، فمثلا القوس الأيمن المربع ']' يكتب بـضغط المفتاحين (Shift) (يـ)، وكذلك الرمز (') يكتب بالضغط على المفتاحين (Shift) (ش).

ويمكن دمج رمزين لتكوين رموز أخرى وهي:

أداة التخصيص : :=

أدوات المقارنة : > < >= <=

محددة المدى : ..

٣ ــ ٢ الكلمات المحجوزة

RESERVED WORDS

الكلمات المحجوزة عبارة عن كلمات عرفت في المترجم لتؤدي هدفا معينا. ولا يمكن تعريفها في البرنامج أو استعمالها لغير غرضها.

منظومة	**عكس**	**أذهب_إلى**	**ملف**
فارغ	**مع**	**أو**	**اعمل**

مجموعة	نص	حيث	م او
سجل	خارجي	إلى	في
سلسلة	مكثف	تنازلي	حتى
من	بداية	باقي	حالة
مطلق	إذا	قص	اجراء
اذن	طالما	برنامج	عند
والا	كرر	عنوان	ثابت
نهاية	امامي	دالة	صنف
متغير	تقسيم	و	

٣ ـ ٣ الأسماء التعريفية

IDENTIFIERS

يستعمل الأسم التعريفي للدلالة على عنوان، وثابت، وصنف، ومتغير، واجراء، ودالة. ويجب أن يبدأ الاسم من حرف هجائي يليه مجموعة رموز قد تتكون من حروف هجائية، أو الرمز (ـــ)، أو أرقام بحيث لا يتعدى طول الاسم ١٢٧ رمزا، ومن الأفضل استخدام اسماء تعريفية مفهومة في البرنامج لتساعد المبرمج على متابعة البرنامج.

أمثلة:

مساحة

مثلث

الأسم___الاول

٤ برتقالات — غير جائز، لأنه يبدأ برقم

قسم الرياضيات — غير جائز، لوجود الفراغ كفاصل

الأسماء التعريفية القياسية

STANDARD IDENTIFIERS

تحتوي لغة نبراس على العديد من أسماء التعريف القياسية المعرفة في المترجم وتشمل ثوابت، ومتغيرات، واجراءات، ودوال، ويمكن تعريف هذه الأسماء في البرنامج واستعمالها لغرض آخر.. ولكنها ستفقد وظيفتها التي خصصت لها في المترجم، وقد تؤدي إلى بعض الالتباسات.

القائمة التالية من الأسماء التعريفية من الأفضل تجنب استعمالها لغير وظيفتها المخصصة لها:

حرف	رقم	انسخ	طول
موقع	وصل	لاحق	سابق
مفتاح_علوي	اهمل___كسر	قرب___كسر	فردي
مطلق	تربيع	جذرت	جا
جتا	قطا	لوغ	قوة
عشوائي	رقم___ص	عشري	عدد_معطيات
معطيات___سلسلة	ادنى	اعلى	بدل
نتيجة___دخ	ن___م	ن___س	بحث___ن ــ م
حجم___ملف	انتظر	موضع___ملف	ضغط___مفتاح
موضع___ملف ــ طويل	سعة___قصوى	سعة_ذم	طول___ذ
مؤشر___ذس___م	اخراج	ادخال	ن___ط
شاشة	ل___م	طباعة	مساعد
مستعمل	اكتب___سطر	اكتب	اقرأ
اقرأ_سطر	احذف	اضف	اذهب___ع___س
خصص	افتح___م___ق	افتح___م___ك	الحق
اغلق	امسح	غير___اسم	بحث___عن

جديد	**علامة**	**اطلق**	**حجز ذاكرة**
احذف___ذاكرة	**ذاكرة___حرة**	**سلسل**	**قيمة**
اسم___دليل	**أول___عشوائي**	**انقل**	**املأ**
اخرج	**قف**	**ارسل___إلى___ملف**	**اهمل**
نفذ	**هيء_شاشة**	**خروج___من___شاشة**	**امسح___شاشة**
امسح___سطر	**لون___اعلى**	**لون___ادنى**	**لون___عادي**
اضف___سطر	**احذف___سطر**	**اقرأ كتلة**	**اكتب___كتلة**

نلاحظ أن بعض الحروف العربية تأخذ رمزين أو أكثر كحرف (ع) مثلا حيث يعتمد نوع الحرف على موقعه في الكلمة، كما أن حرف الألف يأخذ عدة احتمالات، مما يجعل المبرمج يخطيء في كتابة هذه الحروف عند التحرير، وقد تكتب الهاء المربوطة (ه) تاء مربوطة (ة)، وبما أن معظم الكلمات المحجوزة والأسماء التعريفية القياسية تحتوي على تلك الحروف لهذا صمم المترجم على قبول الحروف بأشكالها المحتملة والمتوقعة، كما جعل الرمز (ـــ) مقبولا في أي موقع في الكلمة.

أمثلة:

اكتب ـــ سطر	= أكتبسطر	= اكتبسطر
ضغط_مفتاح	= ضغطمفتاح	= ضغـطمـفتاح
اذهب_ع_س	= أذهبعس	= اذهــبعس
اقرأ	= أقرأ	= اقـــرا

٣ ـ ٥ الثوابت الرمزية

CHARACTER CONSTANTS

الثوابت الرمزية عبارة عن مجموعة متسلسلة من الرموز ومحصورة بين الحاصرتين العلويتين ('').

مثال:

'هذه عبارة عن سلسلة رمزية'

ويمكن أيضا كتابة الحاصة العلوية (') في داخل السلسلة، وذلك بكتابتها مرتين متتاليتين، والسلسلة التي تحتوي على رمز واحد تكون من صنف البيانات (رمز)، كما إن السلاسل متوافقة مع المنظومات التي من نوع (رمز) وتحتوي على نفس العدد من الرموز أو نفس الطول.

أمثلة:

'عبد العزيز '

'صباح ''الخير '

''''

';'

''

في المثال الثاني والثالث تكتب الأداة (') في السلسلة.. أي تعتبر رمزا من رموز السلسلة، والمثال الرابع يبين السلسلة الفارغة.

٣ـ٦ رموز التحكم

CONTROL CHARACTERS

يسمح المترجم باستخدام رموز التحكم داخل السلاسل الرمزية، وتستخدم إحدى الأداتين التاليتين:

١) الأداة = يتبعها عدد صحيح مداه من صفر إلى ٢٥٥، وتمثل الرمز الذي يحصل هذا الرقم في الشفرة (ASCII).

٢) الأداة ^ يتبعها رمزا يعبر عن نوعية رمز التحكم.

أمثلة:

=١٠ ويمثل العدد ١٠ في قائمة الشفرة مفتاح العودة (RETURN)

^G رمز التحكم ^G يجعل الجرس يعمل.

^[رمز مفتاح الهروب (ESC).

كما يمكن عمل سلسلة من رموز التحكم بكتابتها في سلسلة وبدون فواصل بين كل واحدة والأخرى كالآتي:

١٣ = ١٠

٢٧ ^U # ٢٠ ^G^G^G

وتخلط السلاسل الرمزية مع رموز التحكم أيضا مثل:

'أضغط مفتاح العودة للاستمرار ! '^G^G

٣ ــ ٧ الجمل التوضيحية

COMMENTS

تستخدم الجمل التوضيحية في البرنامج لتساعد المبرمج على قراءة وفهم البرنامج. وتستخدم الأقواس المائلة {} لتبين للمترجم الجملة التوضيحية.. وقد تستبدل بالأدوات (* *) لتؤدي نفس الغرض.

أمثلة:

{ هذه جملة توضيحية}

(* وهذه ايضا *)

OOOOOOO

الباب الرابع

التعبيرات EXPRESSIONS

تمهيد:

التعبير وسيلة لبناء حلول العمليات الحسابية، فالعمليات الحسابية عبارة عن معادلات تتكون من رموز وأرقام تؤدي إلى حل عملية معينة. والحاسب الآلي يعتبر من أسرع الوسائل لحل هذه العمليات وخاصة المعقدة منها، ولصياغة المعادلات في لغة البرمجة.. يجب أن نعرف الطريقة الصحيحة لجعل الحاسب الآلي يدركها، وهي مشابهة قليلا للطريقة المتبعة في الرياضيات الحديثة. والطريقة الصحيحة هي وضع المعادلات فيما يسمى بلغة الحاسب الآلي «**التعبير**». ويتكون التعبير من عوامل ومتغيرات، وثوابت، ودوال، تدمج ببعضها البعض لتكون حلا لمعادلة حسابية أو منطقية.

٤ ــ ١ العوامل

OPERATORS

تنقسم العوامل إلى خمسة أصناف، ويمثل الترتيب الآتي أفضلية تنفيذها في التعبير لكل منها:

١) الإشارة السالبة (ــ).

٢) العاملة **عكس**.

٣) عوامل الضرب والقسمة: × ، ÷ ، تقسيم، باقي، و.

٤) عوامل الجمع والطرح: + ، − ، او ، م او

٥) عوامل المقارنة: = ، <> ، < ، > ، >= ، <= ، في.

وعندما يتساوى عاملان في الأفضلية، فإن التنفيذ يتم من اليمين إلى اليسار، والتعبيرات المغلقة بالأقواس تنفذ أولا قبل العوامل. فمثلاً التعبيران التاليان:

(س + ص × ع) = (س + (ص × ع))

عملية الضرب تنفذ أولا في كلا التعبيرين ثم عملية الجمع ولكن

(س + ص × ع) ((س + ص) × ع)

التعبيران لا يتساويان لأن عملية الجمع في الطرف الأيسر تنفذ أولا. وإذا كان كلا طرفي التعبير من صنف (صحيح)، فالنتيجة من صنف (صحيح)، أما إذا كان أحد طرفيها أو كلاهما من صنف (حقيقي)، فالنتيجة من صنف (حقيقي).

٤ ــ ١ ــ ١ الإشارة السالبة UNARY MINUS

تعبر الاشارة السالبة على سلبية العدد سواء كان عددا صحيحاً أو حقيقيا.

٤ ــ ١ ــ ٢ العاملة «عكس»

NOT OPERATOR

تعمل العاملة «**عكس**» على عكس القيمة المنطقية في ظرف العملية فمثلا:

عكس صح = خطأ.

عكس خطأ = صح.

كما يسمح النظام باجراء هذه العملية على الأعداد الصحيحة، فهي تعكس قيم البت (BIT) في البايت.

أمثلة:

عكس ٠ = -١

٠	٠	٠	٠	٠	٠	٠	٠	٠	قيمة الصفر في البايت

خانة الاشارة

١	١	١	١	١	١	١	١	٠	الرقم «١» في خانة الإشارة يعني قيمة سالبة

قيمة عكس الصفر في البايت = -١

حيث تعكس قيم الخانات (BITS) ويضاف إليها الرقم «١».

عكس ١٤ = -١٥

عكس - ٢٠ = ١٩

٤ ـ ١ ـ ٢ عوامل الضرب

MULTIPLYING OPERATORS

القائمة التالية توضح عمليات الضرب والقسمة، وأنواع اطراف العملية وصنف نتيجة العملية.

العامل	العملية	أطراف العملية	نوع النتيجة
×	ضرب	حقيقي	حقيقي
×	ضرب	صحيح	صحيح
×	ضرب	حقيقي، صحيح	حقيقي
÷	قسمة	حقيقي، صحيح	حقيقي
÷	قسمة	حقيقي	حقيقي
÷	قسمة	صحيح	حقيقي
تقسيم (DIV)	قسمة عدد صحيح	صحيح	صحيح
باقي (MOD)	باقي قسمة	صحيح	صحيح
و (AND)	(و) الحسابة	صحيح	صحيح
و	(و) المنطقية	منطقي	منطقي

أمثلة:

١٥ × ١٣	= ١٩٥
١٢٣ ÷ ٤	= ٣٠,٧٥
١٢٣ **تقسيم** ٤	= ٣٠
١٧ **باقي** ٣	= ٢
صح وخطأ	= خطأ
١٢ و ٢٢	= ٤

المثالان الأخيران توضحهما القائمتان التاليتان:

(و) المنطقية

طرف١	طرف ٢	نتيجة
صح	صح	صح
صح	خطأ	خطأ
خطأ	صح	خطأ
خطأ.	خطأ	خطأ

(و) الحسابية

طرف ١	طرف ٢	نتيجة
صفر	صفر	صفر
صفر	واحد	صفر
واحد	صفر	صفر
واحد	واحد	واحد

ونلاحظ إن استعمال (و) الحسابية يجري بعد تحويل العدد الصحيح إلى القيمة الثنائية (BINARY) التي تمثله، ثم اجراء العملية عليه.

٤ ـ ١ ـ ٣ عوامل الجمع

ADDING OPERATORS

القائمة التالية توضح عوامل الجمع، ونوع العملية، وأطرافها، ونتيجتها:

العامل	العملية	أطراف العملية	نوع النتيجة
+	جمع	حقيقي	حقيقي
+	جمع	صحيح	صحيح
+	جمع	حقيقي، صحيح	حقيقي
−	طرح	حقيقي	حقيقي
−	طرح	صحيح	صحيح
−	طرح	حقيقي، صحيح	حقيقي
او (OR)	(أو) الحسابية	صحيح	صحيح
او	(او) المنطقية	منطقي	منطقي
م او (XOR)	(م او) الحسابية	صحيح	صحيح
م او	(م او) المنطقية	منطقي	منطقي

أمثلة:

١٣٣ + ٤٥٠	= ٥٨٣
١٥٦ - ٢٣,٠	= ٢٣٣,٠
صح او خطأ	= صح
١٢ او ٢٢	= ٣٠
صح م او خطأ	= صح
١٢ م او ٢٢	= ٢٦

القوائم التالية توضح الأمثلة الأخيرة من المثال السابق:

(او) المنطقية

طرف١	طرف٢	النتيجة
صح	صح	صح
صح	خطأ	صح
خطأ	صح	صح
خطأ	خطأ	خطأ

(او) الحسابية

طرف١	طرف٢	النتيجة
صفر	صفر	صفر
صفر	واحد	واحد
واحد	صفر	واحد
واحد	واحد	واحد

(م او) المنطقية

طرف١	طرف٢	النتيجة
صح	صح	خطأ
صح	خطأ	صح
خطأ	صح	صح
خطأ	خطأ	خطأ

(م او) الحسابية

طرف١	طرف٢	النتيجة
صفر	صفر	صفر
صفر	واحد	واحد
واحد	صفر	واحد
واحد	واحد	صفر

٤ ــ ١ــ ٤ عوامل المقارنة

RELATIONAL OPERATORS

تعمل عوامل المقارنة على جميع أنواع المقاييس القياسية (حقيقي، صحيح، منطقي، رمز، بايت)، ويمكن خلط الأنواع الثلاثة (صحيح، حقيقي، بايت) في تعبير المقارنة، والنتيجة دائما من نوع منطقي أي «صح» أو «خطأ».

تشمل عوامل المقارنة العوامل التالية:

=	يساوي.
< >	لا يساوي.
<	أكبر من.
>	أقل من.
< =	أكبر من أو يساوي.
> =	أقل من أو يساوي.

أمثلة:

أ =	ب	صح، إذا كان «أ» يساوي «ب».
أ < >	ب	صح، إذا كان «أ» لا يساوي «ب».
أ ><	ب	صح، إذا كان «أ» أكبر من «ب».
أ «	ب	صح، إذا كان «أ» أصغر من «ب».
أ < =	ب	صح، إذا كان «أ» أكبر من أو يساوي «ب».
أ .> =	ب	صح، إذا كان «أ» أصغر من أو يساوي «ب».

٤ ـ ٢ مخصصات الدالة

FUNCTION DESIGNATORS

مخصصات الدالة عبارة عن أسماء تعريفية، يتبعها في بعض الأحيان قائمة معطيات، قد تشمل متغيرا أو أكثر أو تعبيرا آخر، يفصل بينها فاصلة (،)، وموضوعة داخل أقواس. وعند حدوث مخصص الدالة في التعبير.. ينفذ المترجم الدالة التي تحمل هذا الاسم، فإذا كانت غير موجودة في المترجم (داخلية)، فيجب تعريفها في البرنامج قبل وضعها في التعبير.

مثال:

اكتب سطر (ج × (جذرت (ع))

(جذر (س) > ١٥) و (خ) < جا (ق))

○○○○○○○

الباب الخامس

تصميم هيكل البرنامج

PROGRAM STRUCTURE

تمهيد

ان أفضل الوسائل لتعلم البرمجة هي محاولة تصميم البرنامج، ثم ادخاله في الحاسب الآلي وتنفيذه، فالكتب قد تعلم البرمجة ولكن ليست أفضل من التجربة العملية، من الوسائل الحديثة التي تسهل عملية تصميم البرنامج هي تنظيم وتقسيم بنية البرنامج إلى أقسام رئيسية. ففي هذا النظام يقسم هيكل البرنامج إلى ثلاثة أجزاء رئيسية وهي: مقدمة البرنامج، ومنطقة التعريف، ومنطقة العمليات، كما تحتوي هذه المناطق على أجزاء فرعية، ويتكون هيكل البرنامج كما يلي:

مقدمة البرنامج

اسم البرنامج

موجهات البرنامج

منطقة التعريف

العناوين

الثوابت

المصنفات

المتغيرات

تعريف البيانات

منطقة العمليات

اجراءات
دوال ــــ الكتلة الرئيسية
الجمل التنفيذية

٥ ــ ١ مقدمة البرنامج

PROGRAM HEADER

تأتي المقدمة في السطرين الأولين من البرنامج، فالسطر الأول يوضح اسم البرنامج، وقائمة معطيات توضح المحيط الذي يتعامل معه البرنامج، والسطر الثاني يوضح موجهات البرنامج، ووظيفتها التحكم في فحص بعض أنواع الأخطاء والسيطرة على عمليات ادخال واخراج البيانات، وتعتبر اختيارية للمبرمج ولا تعمل أي تأثير على وظيفة البرنامج.

أمثلة:

برنامج تحويل ؛

برنامج حساب (ادخال ، اخراج) ؛

برنامج كتابة (ادخال ، الطابعة) ؛

٥ ــ ٢ منطقة التعريف

DECLARATION PART

تشمل منطقة التعريف على جميع أسماء التعريف المستعملة في منطقة العمليات. وتقسم منطقة التعريف إلى خمسة أقسام أو أجزاء:

١) العناوين.

٢) الثوابت.

٣) التصنيف.

٤) المتغيرات.

٥) الاجراءات والدوال.

٥ ـ ٢ ـ ١ تعريف العناوين

LABELS DECLARATION

تستخدم العناوين في البرنامج لوضع علامة على بعض نقاط البرنامج باستخدام العبارة «اذهب ـ إلى» مع العنوان.. يستطيع المبرمج أن يرغم المترجم على القفز من مكان إلى آخر داخل البرنامج، وتنفيذ بعض العمليات قبل الأخرى. العديد من المبرمجين يعتبرون استعمال العبارة «أذهب ـ إلى» وسيلة ضعيفة للبرمجة.. لأنها تؤدي إلى الإلتباس وعدم التنظيم في البرنامج. تعرف منطقة العناوين بالكلمة المحجوزة «عنوان»، ثم تكتب قائمة العناوين في هذه المنطقة.

مثال:

عنوان

رسالة، انتهى، الخطوة ـ التالية ؛

٥ ـ ٢ ـ ٢ تعريف الثوابت

CONSTANTS DECLARATION

يستخدم المبرمج بعض القيم الثابتة أحيانا في البرنامج، فاستعمال الثوابت يوفر للمبرمج بعض الوقت، ويبسط البرنامج، فإذا أراد أن يغير المبرمج قيمة متغير، فلن يحتاج أن يغير جميع القيم في البرنامج وإنما يغير قيمة الثابت فقط. تعرف منطقة الثوابت بالكلمة المحجوزة «ثابت»، ثم تكتب قائمة الثوابت

وقيمتها المخصصة لها، ويوجد في النظام نوعان من الثوابت: **المصنف والغير مصنف**، والفرق بينهما أن المصنف يحتجز ذاكرة أقل من الغير مصنف عند تنفيذ البنرامج وسوف نوضح خصائص كل منها في الباب الخاص بالثوابت.

أمثلة:

ثابت

ثوابت غير مصنفة:

أيام الاسبوع = ٧ ؛

ساعات اليوم = ٢٤ ؛

رسالة = 'السلام عليكم' ؛

ثوابت مصنفة:

أيام الشهر : صحيح = ٣٠ ؛

وصية : **سلسلة** [٢٢] = 'اهلا وسهلا' ؛

أعلى قيمة : حقيقي = ٩٩٩٩,٩ ؛

٥ ــ ٢ ــ ٣ تعريف المصنفات

TYPE DECLARATION

تعرف المتغيرات في البرنامج بطريقتين. الطريقة **المباشرة** وتأتي في منطقة المتغيرات، وطريقة **الاسناد**، حيث يعرف المتغير مباشرة في منطقة تعريف المصنفات، ثم يسند إليه في منطقة تعريف المتغيرات.

تبدأ منطقة المصنفات بالكلمة المحجوزة «**صنف**»، يليها أسماء تعريفية ومتغيرات مخصصة بعلامة التساوي «=» إلى أصنافها.

مثال:

صنف

سجل شخصي = **سجل**

الاسم: **سلسلة** [٢٠] ؛

العمر: صحيح ؛

الراتب: حقيقي ؛

نهاية ؛

يوم = (سبت، أحد، اثنين، ثلاثاء، أربعاء، خميس، جمعة) ؛

قائمة طلبات = **منظومة** [١..٢٥] **من سلسلة** [١٥] ؛

٥ ـ ٢ ـ ٤ تعريف المتغيرات

VARIABLES DEFINITION

المتغير عبارة عن قيمة متغيرة، فيجب تعريف جميع المتغيرات التي في البرنامج قبل استخدامها.. لكي تكون معروفة عند المترجم، وتعرف المتغيرات في منطقة التعريف التي تبدأ بالكلمة المحجوزة «**متغير**»، يليها متغير أو أكثر يفصل بينها الفاصلة (،)، ثم النقطتان (:) ثم الصنف. يعتبر المتغير «**محلي**» (LOCAL) ومعروف فقط في كتلة البرنامج التي هو فيها، كما أن المتغيرات المعرفة داخل الاجراءات والدوال، غير معروفة خارج كتلتها.. أي أن البرنامج الرئيسي لن يعرف هذه المتغيرات إذا حدثت فيه. كما يعتبر المتغير المعرف في البرنامج الرئيسي متغير «عام» (GLOBAL) ومعروف في جميع الدوال والاجراءات التي في كتلة البرنامج.

مثال:

برنامج عام ؛

متغير

ا، س، ك : صحيح ؛

كتاب = **سجل**

اسمه : **سلسلة** [٢٥] ؛

صفحاته : صحيح ؛

```
نصه : منظومة [١..١٠٠٠] من سلسلة [٢٠] ؛
نهاية ؛
علم: منطقي ؛
اجراء محلي ؛
متغير
ب، ع : حقيقي ؛
بداية
...
...
...
نهاية ؛
بداية [البرنامج الرئيسي]
...
...
...
محلي ؛
...
نهاية .
```

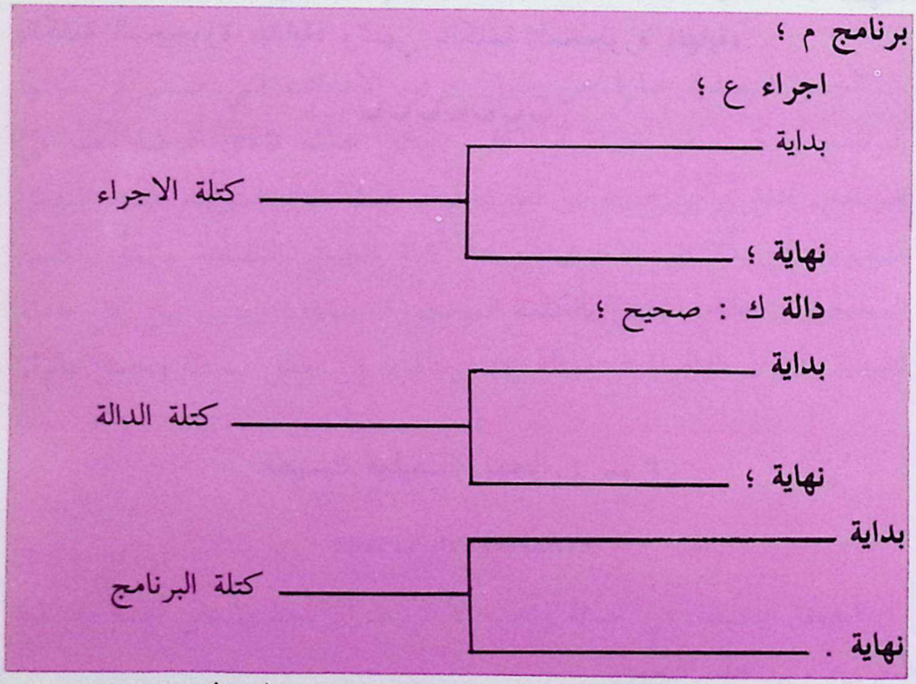
المتغيرات المعرفة في البرنامج الرئيسي عامة على جميع كتلة البرنامج ويمكن استخدامها أيضا داخل الاجراءات والدوال، بينما المتغيرات المعرفة في الاجراء تكون محلية في كتلة الاجراء، فإذا استخدمت في جمل البرنامج الرئيسي فلن تكون معروفة.

٥ ـ ٣ منطقة العمليات

OPERATIONAL PART

منطقة العمليات هي الجزء الرئيسي والأطول في البرنامج، فهي تشمل الاجراءات والدوال والجمل التنفيذية. وتقسم منطة العمليات إلى ثلاث كتل وهي كتلة الاجراء، وكتلة الدالة، وكتلة الجمل التنفيذية.

الشكل التالي يوضح البنية الصحيحة لمنطقة العمليات:



٥ ـ ٣ ـ ١ تعريف الاجراءات والدوال

PROCEDURE AND FUNCTION DECLARATION

الاجراءات والدوال عبارة عن برامج ثانوية تؤدي غرضا معينا في البرنامج فهي مفيدة في العمليات المكررة. وهي مشروحة شرحا مفصلا في الباب الخاص عن الاجراءات والدوال.

٥ ـ ٣ ـ ٢ الجمل التنفيذية

STATMENTS

تأتي الجمل التنفيذية في آخر الكتلة الرئيسية، وتمثل العمليات التي سوف تجري في البرنامج، وتتكون من جملة مركبة أو أكثر، حيث كل جملة تؤدي غرضا معينا في البرنامج، وتنتهي كل جملة بالفاصلة «؛»، وتبدأ هذه الكتلة بالكلمة المحجوزة «**بداية**» وتنتهي بالكلمة المحجوزة «**نهاية**».

OOOOOOO

الباب السادس

الجمل التنفيذية STATEMENTS

تمهيد

الجمل التنفيذية عبارة عن جمل تعرف الأهداف التي صمم من أجلها البرنامج وتتكون من جملة أو اكثر.. وكل جملة تؤدي غرضا معينا في البرنامج، فلغة نبراس عبارة عن لغة تتابعية، حيث يقوم المترجم بتنفيذ الجمل التنفيذية الواحدة تلو الأخرى. وتبدأ كتلة الجمل التنفيذية دائماً بالكلمة المحجوزة «بداية» وتنتهي بالكلمة المحجوزة «نهاية»، ويعزل بين كل جملة تنفيذية وأخرى الفاصلة المنقوطة (؛). وتتكون من جمل بسيطة وجمل بنائية.

٦ ــ ١ الجمل التنفيذية البسيطة

SIMPLE STATEMENTS

الجملة البسيطة هي جملة واحدة لا ترتبط أو تحتوي على جمل إضافية أخرى، ومن هذه الجمل:

١) جملة التخصيص.
٢) جملة الاجراء.
٣) العبارة «اذهب_إلى».
٤) الجملة الفارغة.

٦ ــ ١ ــ ١ جملة التخصيص

ASSIGNMENT STATEMENT

تستخدم جملة التخصيص لتخصيص قيمة معينة إلى متغير معين، وهي أكثر الجمل استعمالاً في البرامج، وتتكون جملة التخصيص من اسم تعريف يليه عامل التخصيص (:=)، ثم تعبير أو اسم تعريف آخر أو قيمة معينة، ويجوز استعمال التخصيص على جميع أنواع المتغيرات ما عدا المتغيرات التي من نوع (الملفات)، ويشترط أن يكون طرفا جملة التخصيص من نوع واحد، باستثناء ما إذا كان المتغير من نوع (حقيقي) والتعبير من نوع (صحيح) فهذا جائز.

أمثلة:

مساحة:= طول × عرض × ارتفاع ؛	{تعبير}
موجود:= خطأ؛	{قيمة منطقية}
الزاوية___اليمنى:= ٦٥؛	{قيمة صحيحة}
م := ن ؛	{اسم تعريف}

٦ ــ ١ ــ ٢ جملة الاجراء

PROCEDURE STATEMENT

تعمل جملة الاجراء على تنفيذ اجراء معين، وقد يكون الاجراء معرفاً بواسطة المبرمج أو اجراء قياسي. وتتكون الجملة من اسم تعريف الاجراء يليه قائمة معطيات اختيارية مغلقة بين قوسين، وتتكون المعطيات من متغيرات أو تعبيرات تفصل الفاصلة (،) بين الواحدة والأخرى. عند حدوث جملة الاجراء في البرنامج خلال التنفيذ، فإن السيطرة الكاملة تتحول إليه، وتمرر المعطيات للاجراء، وعند نهاية تنفيذه يقوم المترجم بتنفيذ الجملة التي تلي جملته.

أمثلة:

ترتيب_تنازلي (اسماء، عدد)؛

عدد_حروف (اسم_ملف)؛

ابحث (اسم، عنوان)؛

٦ ــ ١ ــ ٣ العبارة «اذهب_إلى»

GOTO STATMENT

تتكون العبارة من الكلمة المحجوزة «اذهب_إلى»، يليها اسم تعريف من نوع (عنوان)، وتعمل العبارة على انتقال التنفيذ إلى العنوان المعين في البرنامج، ويجب ملاحظة شرطين أساسيين قبل استعمال هذه العبارة:

١) يجب أن يكون العنوان معرفا من قبل في المنطقة المخصصة لتعريف العناوين في نفس الكتلة التي تستخدم فيها العبارة.

٢) إن حدود العبارة هي كتلتها، وتعتبر غير معروفة خارج مدى هذه الكتلة، فمن المستحيل مثلا الخروج من مدى اجراء معين إلى اجراء آخر بهذه العبارة.

```
برنامج اختياري ؛
عنوان
    ١ ، ٢ ، ٣ ، ٤ ؛
متغير
    اختيار : رمز ؛
بداية
    اكتب (' اختر م ، ن،   ل : ') ؛
    اقرأ_سطر (اختيار) ؛
    حالة اختيار من
      'م' : اذهب الى ١ ؛ {ه نفذ الجملة ١ ه}
```

```
'ن' : اذهب الى ٢ ؛ { نفذ الجملة ٢ }
'ل' : اذهب الى ٣ { نفذ الجملة ٣ }
والا
اذهب الى ٤ ؛
نهاية؛
١ : اكتب_سطر (' الجملة = ١')؛
٢ : اكتب_سطر (' الجملة = ٢ ')؛
٣ : اكتب_سطر (' الجملة = ٣ ')؛
٤ : اكتب_سطر (' ... انتهى ...')؛
نهاية .
```

٦ ـ ١ ـ ٤ الجملة الفارغة

EMPTY STATEMENT

الجملة الفارغة عبارة عن جملة لا تحتوي على أي رمز، وليس لها تأثير في البرنامج وإنما تستعمل لبعض أغراض البرمجة، كجعلها قيمة مبدئية لسلسلة رمز أو استعمالها في تعبيرات المقارنة.

مثـال:

```
بداية
طالما (جواب '' ) اعمل
كرر حتى ضغطمفتاح ؛ (* أنتظر حتى ضغط اي مفتاح *)
نهاية ؛
```

٦ ـــ ٢ الجمل البنائية

STRUCTURED STATEMENTS

الجملة البنائية عبارة عن فقرات تشتمل على جمل تنفيذية، يقوم المترجم بتنفيذ هذه الجمل بطريقة تتابعية، وتتكون الجمل البنائية من جمل مركبة، وجمل شرطية، وجمل التكرار.

٦ ـــ ٢ ـــ ١ الجملة المركبة

COMPOUND STATEMENT

تستخدم الجملة المركبة إذا كان هناك أكثر من جملة تنفيذية واحدة في البرنامج، وتتكون من مجموعة من الجمل التنفيذية يفصل بين كل واحدة والأخرى الفاصلة المنقوطة «؛»، وإذا كانت الجملة المركبة تحتوي على أكثر من جملة تنفيذية واحدة فيحب أن تبدأ بالكلمة المحجوزة «بداية» وتنتهي بالكلمة «نهاية»، وينفذ المترجم الجمل بالترتيب الذي تكتب فيه.

مثـــال:

```
اذا صغير > كبير اذن
بداية
   مؤقت := صغير ؛
   صغير := كبير ؛
   كبير := مؤقت ؛
نهاية ؛
```

جملة مركبة (الأسطر الثلاثة بين «بداية» و«نهاية»)

٦ ــ ٢ ــ ٢ الجمل الشرطية

CONDITIONAL STATEMENTS

تسمى الجمل الشرطية أيضا بالجمل المنطقية، لأن نتيجة تنفيذ هذه الجمل تكون إما صح أو خطأ، وتستخدم الجمل الشرطية إذا أريد تنفيذ بعض الجمل في حالة صحة الشرط فقط، وتنفيذ جمل أخرى في الحالة الأخرى، أو لتنفيذ جمل معينة فقط دون الأخرى.

يوجد في المترجم عبارتان شرطيتان هما:

١) عبارة «**إذا**».

٢) عبارة «**حالة**».

٦ ــ ٢ ــ ٢ ــ ١ العبارة «إذا»

IF STATEMENT

تستخدم العبارة «**اذا**» لتكوين جمل شرطية تحدد تنفيذ بعض الجمل عند حدوث صحة الشرط. ويمكن وضع اختيار آخر عند عدم صحة الشرط، وذلك باستخدام الكلمة المحجوزة «**وإلا**»، ثم كتابة جمل تنفيذية بعدها، وهي اختيارية أي أن العبارة «**اذا**» صحيحة بها أو بدونها. ويجب ملاحظة أن هذه الكلمة لا يجوز أن تسبقها الفاصلة المنقوطة (؛)، ويأتي تنظيم الجملة كالآتي:

الجملة كالآتي:

إذا تعبير_١ اذن

اذا تعبير_٢ اذن

جملة_١

والا

جملة____٢

ويلاحظ أن الكلمة «والا» تتبع لآخر جملة «إذا»، ولأزالة الالتباس يمكن كتابة التنظيم السابق كما يلي:

اذا تعبير_١ **اذن**

بداية

اذا تعبير_٢ **اذن**

جملة_١

والا

جملة_٢

نهاية؛

مثـال:

اذا السن < ١٨ **اذن**

قاصر := صح

والا

قاصر := خطأ ؛

اذا (الوزن> ٥٠) و (الطول > ١٥٠) **اذن**

بداية

اكتب (' الرجاء كتابة الاسم : ') ؛

اقرأ_سطر (اسم) ؛

نهاية ؛

في المثال استعملنا أداة الجمع المنطقية «و» بحيث أن الشرط لا يتحقق ويصبح صحيحا إلا بصحة كلا العبارتين.

ويمكن كتابة العبارة «إذا» بطريقة معششة (NESTED)، فتوضع مجموعة من هذه العبارة داخل بعضها البعض لعمل جمل شرطية معقدة.

```
برنامج كبيسة ؛
متغير
        سنة_كبيسة        : منطقي ؛
        سنة              : صحيح ؛
بداية
   اكتب_سطر (' السنة الكبيسة ' ) ؛
  كرر
    اكتب_سطر ؛
    اكتب (' ادخل السنة (صفر للخروج) : ') ؛
    اقرأ_سطر (سنة) ؛
    اذا سنة> ٠ اذن
    بداية
      إذا سنة باقي ١٠٠ = ٠
       اذن سنة_كبيسة := (سنة باقي ٤٠٠) = ٠
       والا سنة_كبيسة := سنة باقي ٤) = ٠ ؛
      اذا سنة_كبيسة اذن
        اكتب_سطر (سنة''سنة كبيسة')
      والا
        اكتب_سطر (سنة ،' سنة غير كبيسة') ؛
      نهاية {اذا}
    حتى سنة = ٠
  نهاية .
```

٦ ـ ٢ ـ ٢ ـ ٢ العبارة «حالة»

CASE STATEMENT

تستخدم العبارة «**حالة**» لتكوين جملة شرطية اختيارية، وتتكون من تعبير يلحقه قائمة جمل تحمل جميع الإحتمالات والاختيارات التي قد يحققها الشرط، وكل إحتمال يحمل عنوانا يمثل قيمة محتملة لتحقيقه، وقد يشمل جملا تنفيذية تجعل المترجم ينفذ أهدافا مختلفة عن الإحتمالات الأخرى، وإذا لم تتحقق الشروط فالمترجم لن ينفذ أي جملة، إلا إذا وجدت الكلمة المحجوزة «**والا**» تليها جمل تنفيذية، فبذلك يعمل المترجم على تنفيذ الجمل في حالة عدم توافق الشرط مع جميع الإحتمالات، ويجب وضع الكلمة «**نهاية**» لتشير إلى نهاية قائمة الإحتمالات بعد آخر إحتمال. ومن الأفضل استعمال هذه العبارة بدلا من «**إذا**» إذا كان هناك العديد منها، لتجعل قراءة وفهم البرنامج بسيطة وسهلة.

ويتكون التعبير من الكلمة المحجوزة «**حالة**» يتبعها متغير يسمى **متغير الاختيار** ثم الكلمة المحجوزة «**من**». أما عنوان إحتمالات العبارة «**حالة**» فيتكون من اي عدد من الثوابت أو مدى فرعي يفصل بينها الفاصلة وتتبعها النقطتان (:)، ويجب أن تكون جميع الإحتمالات من صنف واحد.

أمثلة:

حالة اداة **من**

'+' : نتيجة := مجموع + نتيجة ؛

'−' : نتيجة := مجموع − نتيجة ؛

'×' : نتيجة := مجموع × نتيجة ؛

'÷' : نتيجة := مجموع ÷ نتيجة ؛

نهاية ؛ (٭ حالة ٭)

```
حالة مفتاح من
 'ا'.. 'ي':
                    اكتب_سطر (' هذا مفتاح حرف هجائي ') ؛
 '٠'.. '٩':
                    اكتب_سطر (' هذا مفتاح رقم حسابي ')
والا
  بداية
    اكتب_سطر (' هذا مفتاح غير معروف ') ؛
    اكتب_سطر (' حاول مرة اخرى !!! ') ؛
  نهاية ؛
نهاية ؛
```

البرنامج الآتي يوضح طريقة استعمال العبارة «**حالة**» في البرنامج:

```
برنامج الوان ؛
متغير
  اختيار : رمز ؛
  انتهاء : منطقي ؛
بداية
  اكتب_سطر (' الوان الطيف') ؛
  انتهاء := خطأ ؛
  كرر
    اكتب (' أ (ز)رق، أ (خ)ضر، (ب)رتقالي، (و)ردي، أ(ح)مر،
          أ(ص) فر (ن)هاية ') ؛
    اقرأ_سطر (اختيار) ؛
    حالة اختيار من
      'ز' : اكتب_سطر (' أزرق ') ؛
```

```
'خ' : اكتب_سطر ('أخضر ') ؛
'ب' : اكتب_سطر ('برتقالي ') ؛
'و' : اكتب_سطر ('وردي ') ؛
'ح' : اكتب_سطر ('أحمر ') ؛
'ص' : اكتب_سطر (' أصفر ') ؛
'ن' : انتهاء := صح
والا
    اكتب_سطر (' *** خطء في الادخال حاول مرة أخرى ***')
نهاية؛
حتى انتهاء؛
نهاية .
```

٦ ـ ٢ ـ ٣ جمل التكرار

REPETITIVE STATEMENT

تعني جملة التكرار أن المترجم يقوم بتنفيذ جمل معينة بصورة مكررة. حيث أن غالبية لغات البرمجة الحديثة توفر للمبرمج وسيلة من وسائل التكرار، فبدلا من جعل المبرمج يكتب جملا تنفيذية عديدة ومكررة، فان جملة التكرار توفر بعض الوقت وتختصر من طول البرنامج، فإذا كان مرات التكرار معروفا قبل التنفيذ، فان العبارة «**حيث**» هي العبارة المناسبة لهذه الحالة، وإلا فيجب استعمال العبارة «**طالما**» أو العبارة «**كرر**» لعمل جملة التكرار.

٦ ـ ٢ ـ ٣ ـ ١ العبارة «حيث»

FOR STATEMENT

تعتبر العبارة «حيث» من عبارات التكرار القوية في اللغة، فعند استعمال هذه العبارة يجب تحديد نقاط بداية التكرار ونهايته مستخدما متغيرا يسمى متغير

التحكم يعمل كعداد لمرات التكرار، ويمكن تنظيم العداد ليعمل تنازليا أو تصاعديا حسب الوضع المناسب. فإذا أردنا عمل تكرارا تصاعديا فنستخدم العبارة «إلى» بين نقطة البداية والنهاية أما إذا أردنا عمل تكرار تنازلي.. فنستخدم العبارة «**تنازلي**» بين نقطة البداية والنهاية.

ويجب أن تكون قيم البداية والنهاية ومتغير التحكم من نفس النوع وتعتبر جميع الأنواع القياسية البسيطة مقبولة إلا النوع (حقيقي).. فلا يجوز استعماله.

ويأتي تركيب العبارة «حيث» في البرنامج كالآتي:

حيث م := ادنـــى_عدد **إلى** أعلـــى_عدد **اعمل**

بداية

جملة_١ ؛

جملة_ن ؛

نهاية ؛

مثـال :

حيث ع:= ١ **إلى** ١٠ **اعمل**

بداية

اكتب (' ما هو وزنك بالكيلوغرام؟ ') ؛

اقرأ_سطر (وزن) ؛

حالة وزن **من**

'٣٠' .. '٤٥' : اكتب_سطر (' أنت من الوزن الخفيف ')؛

'٤٦' .. '٦٥' : اكتب_سطر (' أنت من الوزن المتوسط ')؛

'٦٦' .. '١٠٠' : اكتب_سطر (' أنت من الوزن الثقيل ')؛

```
والا
    اكتب_سطر (' انت من فئة أوزان غير معروفة ') ؛
نهاية ؛
نهاية ؛
```

٦ ــ ٢ ــ ٣ ــ ٢ العبارة «طالما»

WHILE STATEMENT

تختلف هذه العبارة قليلا عن العبارة «**حيث**»، فهي تحتوي على تعبير يتحكم في التكرار ونتيجة هذا التعبير دائما نتيجة منطقية، ويعتبر التكرار ساريا ما دامت نتيجة هذا التعبير صحيحة، وإذا كانت النتيجة خطأ حتى ولو أن التكرار لم يبدأ فإن المترجم لن ينفذ عملية التكرار.

مثـال:

```
اقرأ (عدد)؛
طالما عدد > ١ اعمل
بداية
    عدد := جذرت (عدد) ؛
نهاية؛
    اكتب_سطر (عدد)؛
اقرأ_سطر (درجة) ؛
طالما درجة <> ٩٩٩ اعمل
بداية
    درجة : = (درجة × ٩ ÷ ٥) + ٣٢ ؛
    اكتب_سطر (' الدرجة بالفهرنهايت = '، درجة) ؛
نهاية ؛
```

يجب ملاحظة العبارة «**طالما**» «لن تعمل إلا باعطاء قيمة مبدئية لتقييم التعبير قبل بداية التكرار، فمثلا في المثال السابق إذا لم تكن هناك قيمة مبدئية للمتغير

«عدد» فإن المترجم لن يستطيع تقييم العبارة ولا يعرف إذا كان «عدد» أكبر من واحد أم لا.

٦ ـ ٢ ـ ٣ ـ ٣ العبارة «كرر»

REPEAT STATEMENT

تشابه العبارة «**كرر**» العبارة «**طالما**»، حيث أنها تحتوي على تعبير منطقي يتحكم في عملية التكرار، إلا أن عملية تقييم هذا التعبير تأتي في النهاية بدلا من البداية، كما أن التكرار يكون مستمرا ما دامت نتيجة التعبير خطأ بدلا من صح في العبارة «**طالما**»، ويسبق هذا التعبير الكلمة المحجوزة «**حتى**». ومعنى هذه العبارة أن التكرار يستمر حتى تصبح نتيجة التعبير صحيحة.

ويأتي تركيب العبارة «**كرر**» كالآتي:

```
كرر
    جملة ـ ١ ؛
        .
        .
        .
    جملة ـ ن ؛
حتى (تعبير منطقي) ؛
```

مثـال:

```
كرر
    اكتب (' ادخل رقم : ') ؛
    ع := ع + ١ ؛
    اقرأ_سطر (رقم [ع]) ؛
    نتيجة := نتيجة + رقم [ع] ؛
حتى (ع = ١٥) او (رقم [ع] = ٠) ؛
```

الباب السابع

المقـايـيس SCALER TYPES

المقاييس عبارة عن مجموعة خطية محدودة من القيم التي تحمل اسما جماعيا واحدا ومرتبة بشكل متسلسل. فكل متغير أو معلومة موجودة في البرنامج يجب أن تكون مصنفة إلى نوع من أنواع المقاييس، ففي هذا الباب سنوضح فئتين من فئات المقاييس: **المقاييس القياسية** وهي بسيطة ومبنية في نظام لغة نبراس، **والمقاييس المعرفة** التي تعرف بواسطة المبرمج في البرنامج.

٧ ــ ١ المقاييس القياسية

STANDARD SCALERS

المقاييس القياسية عبارة عن أنواع بسيطة من البيانات تبين نوعية المتغيرات المستخدمة في البرنامج. فكل متغير يجب أن يمثل ويرتبط بنوع واحد من أنواع البيانات، وهذه الفئة هي المستخدمة بكثرة في البرامج، ويوفر النظام ستة أنواع مصممة في المترجم، وتشمل الأعداد الصحيحة، والأعداد الحقيقية، ووحدة الذاكرة البايت (BYTE)، والقيم المنطقية، والرموز.

٧ ــ ١ ــ ١ الأعداد الصحيحة

INTEGERS

تعرف الأعداد الصحيحة في البرنامج بالكلمة المحجوزة **صحيح**، وهي محدودة المدى فتبدأ من العدد ــ ٣٢٧٦٨ إلى العدد + ٣٢٧٦٨ فالعدد

الصحيح يحتجز وحدتي تخزين في الذاكرة، فاذا تعدى العدد هذا المدى فسيحدث خطأ في التنفيذ.

مثال:

عدد ، ر ، نتيجة : صحيح ؛

٧ ـ ١ ـ ٢ البايت

BYTE

البايت عبارة عن فرع من المقياس القياسي (صحيح).. أي مقياس مدى فرعي، وهي من وحدات قياس الذاكرة في الحاسب الآلي، ومداها يبدأ من صفر إلى ٢٥٥ وهو المدى الذي تشغله وحدة ذاكرة واحدة أو بايت. يعرف المقياس بالكلمة المحجوزة «**بايت**»، ويمكن تخصيصه إلى متغيرات من صنف (صحيح) والعكس أيضا جائز. وتحتوي البايت على ثمانية خانات منطقية، كل خانة تسمى بت (BIT)، وهي أصغر وحدة ذاكرة في الحاسب الآلي، وتأخذ إحدى القيمتين (٠، ١).

الشكل التالي يوضح هيكل البايت وطريقة حساب قيمتها:

٠	٠	٠	٠	١	٠	١	١

$$(١\times٢^{٠}) + (١\times٢^{١}) + (٠\times٢^{٢}) + (١\times٢^{٣}) + (٠\times٢^{٤}) + (٠\times٢^{٥}) + (٠\times٢^{٦}) + (٠\times٧^{٧})$$

$$= ١ + ٢ + ٠ + ٨ + ٠ + ٠ + ٠ + ٠ = ١١$$

مثال:

م، ب : بايت ؛

٧ ـــ ١ ـــ ٣ الأعداد الحقيقية

REAL NUMBERS

تعرف الأعداد الحقيقية بالكلمة المحجوزة **حقيقي**، ومداها أكبر من الأعداد الصحيحة فيبدأ من العدد ــ ٣٨ ق١ إلى العدد +٣٨ ق١، ويرمز الرمز (ق) إلى قوة العدد وهي إحدى عشر رقما كحد أقصى.. وتعني أن العدد إلى يسار (ق) سالب أو موجب مضروب في ١٠ مرفوعة إلى القوة التي على يمين (ق). ويحتجز العدد الحقيقي ستة وحدات (بايث) في الذاكرة، والعمليات الحسابية التي تتعدى المدى سوف تسبب خطأ في التنفيذ وفائض في العدد، مما يؤدي إلى إيقاف تنفيذ البرنامج.

أمثلة:

٥,٣ ، ١٢٣,٥٣٤٨٥

+٦ق ٤,٥ ، +١٢ق -١٤,٩

ويجب أن نلاحظ بعض النقاط والعمليات التي لا يجوز اجراؤها على الأعداد الحقيقية وهي:

١) الدالتان «سابق» و«لاحق» لا تأخذان أعدادا حقيقية كمعطيات.
٢) لا يمكن استعمال الأعداد الحقيقية كفهرس للمنظومات.
٣) المجموعات الأساسية لا تأخذ الأعداد الحقيقية كعنصر من عناصرها.
٤) الأعداد الحقيقية لا تستخدم للتحكم في عبارتي «حالة» و «حيث».
٥) لا يجوز استعمال مقاييس مدى فرعية من الأعداد الحقيقية.

٧ ـــ ١ ـــ ٤ القيم المنطقية

BOOLEANS

تعرف القيم المنطقية بالكلمة المحجوزة «**منطقي**»، وتأخذ القيمة المنطقية

قيمتين وهما «صح» أو «خطأ»، وتأتي القيمة «خطأ» على أنها أقل من القيمة «صح» عند مقارنتهما.. كما تحتجز القيمة المنطقية بايت واحدة فقط من الذاكرة.

مثال:

موجود : منطقي ؛

٧ ـ ١ ـ ٥ الرموز

CHARACTERS

تعرف الرموز بالكلمة المحجوزة «رمز»، وتمثل رمزا واحدا من مجموعة رموز الشفرة (ASCII)، فقد يكون الرمز رمز تحكم، أو رمز خاص، أو حرف عربي، أو حرف انجليزي، أو رقم، أو رمز رسم، وهي مرتبة حسب ترتيبها في قائمة الشفرة، كما يمكن مقارنتها ببعضها ـ فمثلا نجد أن الرمزين ('لا' <'ك')، فنقارن رقميهما في القائمة نجد أن رقم الرمز 'لا' أقل من رقم الرمز 'ك'، ويحجز كل رمز بايت واحد من الذاكرة.

مثال:

ح١ ، ح٢ : رمز ؛

٧ ـ ٢ المقياس المعرف

DECLARED SCALER TYPE

ويعرف المقياس المعرف بأنه يرصد جميع القيم المحتملة مصفوفة بترتيب معين. وتتميز هذه القيم بأنها مجموعة تتكون من نوع واحد ويحمل اسما جماعيا يدل عليها.

أمثلة:

صنف

فصل = (صيف، شتاء، خريف، ربيع) ؛

كلية = (هندسة، علوم، آداب، تربية، شريعة) ؛

يوم = (السبت، الأحد، الأثنين، الثلاثاء، الأربعاء، الخميس، الجمعة) ؛

ويعرف المقياس القياسي (منطقي) كالآتي:

صنف

منطقي = (صح، خطأ) ؛

كما يمكن استخدام عوامل المقارنة = ، >، <، = >، = <، <> في مقارنة عناصر المقياس ببعضها، بشرط أن تكون أطراف المقارنة من صنف واحد، غير أنه يمكن الجمع في مقارنة واحدة بين الصحيح والحقيقي. وترتيب عناصر المقياس في التعريف يمثل الأساس لمقارنتها فاذا رجعنا للمثال السابق نرى أن الترتيب في المقياس (يوم) يبين لنا أن:

السبت < الاحد < الاثنين < الثلاثاء < الأربعاء < الخميس < الجمعة

الدالتان «سابق»، «لاحق» يمكن استخدامهما مع المقياس كالآتي:

لاحق (الاربعاء) لاحق الاربعاء = الخميس

سابق (الاربعاء) سابق الاربعاء = الثلاثاء.

رقم (الاربعاء) رقم الاربعاء = ٤ (حيث رقم السبت = صفر).

صنف نتيجة الدالتان «سابق» و «لاحق» هو من نفس صنف المعامل، أما صنف نتيجة الدالة «رقم» فهو من صنف (صحيح)، ويجب ملاحظة أن عناصر المقاييس لا يمكن استخدامها في الاجرائين (اكتب، اكتب_سطر).

٧ ـ ٣ مقياس المدى الفرعي

SUBRANGE TYPE

مقياس المدى الفرعي هو فرع من مقياس معرف سبق تعريفه، أو فرع من مقياس قياسي. ويعرف مقياس المدى الفرعي بوضع الحد الأدنى والحد الأقصى في القيم، فالقيمة الأولى تمثل الحد الأدنى أما القيمة الأخرى فتمثل الحد الأقصى ويجب أن يكون الحد الأدنى أقل من الحد الأقصى. كما أن الصنف (حقيقي) غير مسموح به في تعريف هذا المقياس.

أمثلة:

قوس_قزح	=	(أزرق، أخضر، أحمر، بنفسجي، أبيض أصفر، زهري) ؛
ورد	=	(أحمر.. أصفر) ؛
درجة	=	٠ .. ٣٦٠ ؛
حروف	=	'أ' .. 'ي' ؛

فمقياس المدى الفرعي (ورد) عبارة عن جزء من المقياس المعرف (قوس_قزح)، ويسمى بذلك المقياس المعرف **بمقياس الارتباط** (THE ASSOCIATED SCALER TYPE). فمقياس الارتباط في مقياس المدى الفرعي (درجة) هو المقياس القياسي **صحيح**، ومقياس الارتباط في (حروف) هو المقياس **رمز** .كما أن هناك مقياس مدى فرعي قياسي وهو **بايت**، ويعرف كالآتي:

صنف

بايت = ٠...٢٥٥ ؛

مقياس المدى الفرعي يحمل جميع صفات مقياس الارتباط، ما عدا أن قيمة مدى كل منها مختلف.

استخدام المقياس المعرف ومقياس المدى الفرعي له فوائد كثيرة، فهو يجعل قراءة البرنامج أكثر وضوحا، بالإضافة إلى توفير مقدار من الذاكرة حيث أن المترجم يحجز بايت واحدة لمتغير المقياس، كما يمكن أن يحتوي المقياس على ما مجموعه ٢٥٦ عنصرا كحد أقصى.

٧ ــ ٤ تحويل صنف قيم المقياس

TYPE CONVERSION

الدالة «**رقم**» تحول صنف قيمة المقياس إلى صنف (**صحيح**) وبهذا النمط يمكن استخدام قيمة المقياس لاستنتاج قيمة مقياس من صنف آخر يحمل نفس الترتيب.. أي نستخدم المقياس كدالة تحمل معطيا واحدا بين قوسين. هذا المعطى قد يكون من أي صنف، ما عدا الحقيقي فلا يجوز استخدامه. وعند الرجوع إلى الأمثلة نجد أن:

صحيح (آداب)	= ٢	ترتيب آداب في المقياس = ٢
يوم (٣)	= الاحد	اليوم الثالث في المقياس (يوم)
قوس_قزح (٤)	= أبيض	اللون الرابع في المقياس (قوس___قزح)
صحيح ('م')	= ١٧٢	رقم الرمز (م) في الشفرة.
حرف (٢٠٥)	= ط	رمز العدد (٢٠٥) في الشفرة.

```
برنامج_قوس_قزح ؛
صنف
  الوان = (أحمر، أخضر، أزرق، أصفر، بنفسجي، أبيض، زهري) ؛
متغير
  لون : الوان ؛
بداية
  امسح_شاشة ؛
  اكتب_سطر ('ترتيب الألوان في المقياس : ') ؛
  اكتب_سطر ؛
  اكتب_سطر (' الدالة (رقم)          الدالة (صحيح) ') ؛
  حيث لون:= أحمر إلى زهري اعمل
  (* اكتب اللون في ٥ خانات وترتيبه في ٢٠ خانة *)
  اكتب_سطر (رقم (لون) : ٥ ، صحيح (لون) : ٢٠) ؛
نهاية .
```

الباب الثامن

السلاسل الرمزيـة STRINGS

يتوفر في لغة نبراس سهولة تداول النصوص، فإذا أردنا ادخال أو اخراج مجموعة أسماء مثلا فإنه من السهولة قراءة الاسم كمتغير كامل بدلا من قراءة كل رمز في الاسم على حدة، ويطلق على هذا النوع من المتغيرات بالسلاسل الرمزية.. وهي عبارة عن سلسلة من الرموز، أما أنواع السلاسل الرمزية فهي أنواع مصممة وشبيهة إلى حد ما بمنظومة من صنف رمز، إلا أن هناك اختلافا رئيسيا وهو أن طول السلسلة متغير، بينما طول المنظومة ثابت لا يتغير.

٨ ــ ١ تعريف سلسلة الرموز في البرنامج

STRING TYPE DEFINITION

عند تعريف سلسلة الرموز يجب تحديد الحد الأعلى من طولها فقط، ويحتوي التعريف على الكلمة المحجوزة «**سلسلة**» وتأتي في منطقة التصنيف أو في منطقة المتغيرات يليها طول السلسلة بين أقواس مربعة، ويجب تحديد الطول بعدد صحيح من ١ إلى ٢٥٥ ، كما يمكن تعريف السلسلة في منطقة الثوابت إذا كان هناك سلسلة معينة تستخدم بكثرة داخل البرنامج:

مثال:

```
ثابت
    رسالة          : سلسلة [٨]= 'اهلا وسهلا'؛
```

```
صنف
    سطر_نص          = سلسلة [٨٠] ؛
    خانة_اسم        = سلسلة [٣٠] ؛
متغير
    سطر             : سطر_نص ؛
    الاسم_الأول      : خانة_اسم ؛
    العنوان          : سلسلة [٤٠] ؛
```

متغيرات السلاسل تحتل الحد الأعلى من طول سلسلة الرموز بالإضافة إلى بايت واحدة إضافية تحتوي على طول المتغير الحالي. البايت الأولى تحتوي على طول السلسلة يليها الرموز مخزنة ومفهرسة حسب ترتيبها في السلسلة فمثلا إذا عرفنا سلسلة بأنها تتكون من خمسة رموز (سلسلة : [٥]) فذلك يعني أنها مخزنة في ستة بايتات.

٨ ــ ٢ التعبير عن السلاسل الرمزية

STRING EXPRESSION

عند معالجة سلسلة الرموز يجب التعبير عنها، والقصد بالتعبير عنها هو وضعها في تعبير يتكون من سلسلة ثابتة أو متغيرة أو مخصصات للدالة.. واستخدام بعض العوامل القياسية معها كعامل الجمع (+) وعوامل المقارنة = ، <، >، <>، >=، <= فيمكن المقارنة بين سلسلة وأخرى من نفس الطول أو من أطوال مختلفة، وتتم المقارنة من اليمين إلى اليسار طبقا لرقم الشفرة القياسية (ASCII) وتكون نتيجة المقارنة منطقية أي «**صح**» أو «**خطأ**».

أمثلة:

'أ' < 'ي'	صح لأن رقم الشفرة في 'أ' أقل من 'ي'
'٧' < '٩'	صح
'خالد' = 'خالد'	صح
'احمد' = 'احمد '	خطأ.

٨ ــ ٣ تخصيص سلاسل الرموز

STRING ASSIGNMENTS

يستخدم عامل التخصيص أو التعيين (:=) لتخصيص قيمة عبارة السلسلة إلى متغير السلسلة. ويستخدم عامل الجمع (+) للربط بين سلسلتين وتخصيصهما إلى متغير سلسلة واحدة.

أمثلة:

شهادة	:= 'لا اله الا الله محمد رسول الله'؛
الاسم_الأول	= 'عمر'؛
الاسم_الثاني	:= 'عبد العزيز'؛
الاسم :=	الاسم الأول + الاسم الثاني + 'عبد الله'؛

إذا كان تعريف طول السلسلة أقل من مجموع الرموز المعطاة فإن الرموز الزائدة سوف تحذف.

٨ ــ ٤ اجراءات سلاسل الرموز

STRING PROCEDURES

لقد صممت في لغة نبراس مجموعة اجراءات قياسية يمكن استخدامها مباشرة في البرنامج لتوفر على المبرمج بعضا من الجهد في تداول السلاسل الرمزية وهي:

احذف (س،م، ع) ؛

هذا الاجراء يحذف من السلسلة (س) عددا من الرموز (ع) ابتداء من الموقع (م). ويكون تعريف (س) بمتغير سلسلة، أما (م) و(ع) فإنهما عددان صحيحان.

مثـال:

إذا فرضنا ان س = 'صباح الخير' فان الاجراء.

أحذف (س ، ٣ ، ٢) سوف يجعل س = 'صب الخير'.

ويجب ملاحظة النقاط التالية عند استخدام هذا الاجراء:

١) إذا كان (م) أكبر من طول السلسلة، فالاجراء لن يحذف شيئاً.

٢) إذا كان (م) + (ع) يتعدى على طول السلسلة، فان الرموز التي داخل السلسلة هي المحذوفة فقط.

٣) إذا كان (م) خارج المدى المسموح به ١..٢٥٥ فسوف يحدث خطأ في التنفيذ.

أضف (س١، س٢، م) ؛

أضف السلسلة (س١) إلى السلسلة (س٢)، بداية عن الموقع (م) في السلسلة (س٢). (س١) يجب أن تكون سلسلة رمزية، (س٢) متغير سلسلة، (م) عدد صحيح.

مثـال:

إذا فرضنا ان س٢ = 'طير السماء' فان الاجراء

أضف ('في'، س٢، ٥) سوف يجعل س٢ = 'طير في السماء'.

يجب ملاحظة الآتي عند استخدام هذا الاجراء:

١) إذا كان (م) أكبر من طول السلسلة (س٢) فان (س١) لن تتم إضافتها.
٢) إذا كانت النتيجة أطول من الحد الأعلى لطول السلسلة (س٢) فإن الزيادة سوف تحذف.
٣) إذا وقع خارج المدى المسموح به ١..٢٥٥ فسوف يحدث خطأ في التنفيذ.

سلسل (ع، س) ؛

الاجراء يحول العدد الصحيح أو الحقيقي (ع) إلى السلسلة (س). ويمكن تشكيل صياغة (س) باستعمال نفس صياغة الاجراء (اكتب).

مثال:

لو فرضنا ان ع = ١٥٦٧٤ فإن لاجراء

سلسل (ع: ٦، س) يجعل س = '١٥٦٧٤'

ولو فرضنا ان ع = ٤+ق ١,٥ فان الاجراء

سلسل (ع: ١٠: ٠ ، س) يجعل س = '   ١٥٠٠٠'

قيمة (س، ع، ك) ؛

هذا الاجراء، هو نقيض الاجراء السابق، يحول عبارة السلسلة (س) إلى عدد صحيح أو حقيقي ثم يخزنه في (ع) ، و يجب أن تكون (س) سلسلة تحتوي على قيمة عددية، كما أن الفراغات غير مسموح بها. اما (ك) فهو عدد صحيح يصبح صفرا إذا كان الاجراء صحيحا، ويعطي موقع الخطأ إذا كان هناك خطأ في الاجراء وتصبح قيمة (ع) غير معرفة.

أمثلة:

١) س = '٥٤٧'

قيمة (س، ع، ك) يعطي (ع) القيمة ٥٤٧ ويضع صفر في (ك).

٢) س = '٥٢ج ٣٦'

قيمة (س، ع، ك) يعطي (ع) قيمة غير معرفة ويضع ٣ في (ك).

٣) س = '٤ + ق١,٦' = ١,٦ × ١٠^٤ = ١٦٠٠٠

قيمة (س، ع، ك) يعطي (ع) القيمة ١٦٠٠٠ ويضع صفر في (ك).

٨ــ٥ دوال سلاسل الرموز

STRINGS FUNCTIONS

تحتوي لغة نبراس أيضا على دوال قياسية داخلية وهي:

انسخ (س، م، ع)؛

هذه الدالة تنسخ من السلسلة (س) بداية من الموقع (م) الرموز التي عددها (ع)، حيث تعرف (س) على أنها تعبير سلسلة أما (م) و (ع) فهما عددان صحيحان.

مثال:

لو فرضنا ان س = 'العلم نور' فإن الدالة

انسخ (س، ٣، ٣) = 'علم'

انسخ (س، ٧، ٦) = 'نور'

يجب ملاحظة الآتي عند استخدام هذه الدالة:

١) اذا كان (م) أكبر من طول السلسلة (س) فإن الدالة تعطي سلسلة فارغة.

٢) اذا كان (م) + (ع) أكبر من طول السلسلة (س) فإن الرموز التي في السلسلة فقط هي التي تعيدها الدالة.

٣) اذا وقع المتغير (م) خارج المدى المسموح به **١..٢٥٥** فسوف يحدث خطأ في التنفيذ.

وصل (س١، س٢، ...)؛

الدالة تعطي السلسلة التي تربط بين (س١) و (س٢) أو أي عدد محدود من السلاسل الرمزية، بشرط أن لا يكون طول السلسلة الناتجة أكثر من ٢٥٥ رمز والا سوف يحدث خطأ في التنفيذ، أيضا يمكن استعمال عامل الجمع (+) لأداء نفس الغرض كما شرحنا سابقا.

مثال:

س١ = 'يوم' 'س٢ = 'غائم'

وصل (س١، 'امس'، س٢) = 'يوم امس غائم'

طول (س)؛

تعطي طول السلسلة (س) أو عدد الرموز الموجودة في (س)، وتكون النتيجة مصنفة كعدد صحيح.

مثال:

س='سنة ١٩٧٥'

طول (س) = ٨

موقع (س١، س٢)؛

تعمل هذه الدالة على فحص السلسلة (س٢) إلى أن تجد بداية حدوث السلسلة (س١) فيها، ونتيجتها من صنف عدد صحيح وهو موقع مطابقة أول رمز من (س١) في السلسلة (س٢). اذا كانت النتيجة صفر فإن السلسلة (س١) غير موجودة في السلسلة (س٢).

مثال :

اذا كان س١ = 'العلم نور' فإن الدالة

موقع ('نور' ، س٢) =٧.

موقع ('فجر' ، س٢) تعطي النتيجة صفر.

```
برنامج البحث_عن_سلسلة؛
صنف
    اكبر_سلسلة = سلسلة [١٣٢]؛

متغير
    اسم_ملف : سلسلة [١٥]؛
    متغير_ملف : نص؛
    رقم_السطر : صحيح؛
    سطر_واحد، مؤقت؛ جزء_السلسلة : اكبر_سلسلة؛
(٭ يبحث عن سلسلة رمزية معينة في الملف المطلوب ثم يعطي السطر الذي فيه السلسلة ٭)
```

```
بداية
        امسح_شاشة؛
        اكتب (' ماهو اسم الملف المراد البحث فيه؟')؛
        اقرأ_سطر (اسم_ملف)؛
        خصص (متغير_ملف، اسم_ملف)؛
        كرر
            اكتب_سطر؛
            افتح_م_ق (متغير_ملف)؛
            اكتب ('البحث عن السلسلة (اضغط مفتاح العودة للانتهاء):')؛
            اقرأ_سطر (جزء_السلسلة)؛
            اكتب_سطر؛
            اذا طول (جزء_السلسلة) > ٠ اذن
            بداية
                رقم_السطر :=٠؛
                (٭ ابحث بالملف حتى نهاية ٭)
                طالما عكس ن م (متغير_ملف) اعمل
                بداية
                    اقرأ_سطر (متغير_ملف، سطر_واحد)؛
                    رقم_السطر := رقم_السطر + ١؛
                    مؤقت := سطر_واحد؛
                    اذا موقع (جزء_السلسلة، مؤقت) >٠ اذن
                        اكتب_سطر (رقم_السطر : ٣، ' :' ، سطر_واحد)؛
                نهاية؛ (٭ طالما ٭)
            نهاية؛ (٭ اذا ٭)
        حتى طول (جزء_السلسلة) =٠؛
            اغلق (متغير_ملف)
نهاية.
```

الباب التاسع

المنظومـات ARRAYS

تمهيد

يحتاج المبرمج إلى نوع من البنية الهيكلية في اللغة لتنظيم ومعالجة العديد من المتغيرات دفعة واحدة، فإذا كان البرنامج سيحتوي على متغيرات كثيرة ومن صنف واحد فإن عملية تداولها وتخزينها تكون شاقة والمنظومات احدى الوسائل الهيكلية التي يوفرها نظام لغة نبراس لتسهل هذه العمليات.

المنظومة نوع مصمم في نظام اللغة ليحتوي على عدد محدود من الخانات التي تخصص لتخزين عناصر من صنف واحد. ويمكن معاملة أي خانة كخانة مستقلة داخل المنظومة، حيث ان كل منها لها فهرس معين، وتكون المنظومة ذات بعد واحد وتسمى في الرياضيات «**متجه**»، أو ذات بعدين وأكثر وتسمى «**مصفوفة**».

٩ــ١ تعريف المنظومات في البرنامج

ARRAYS DEFINITION

تعرف المنظومة داخل البرنامج بالكلمة المحجوزة «**منظومة**» ، يتبعها الحد الأدنى والحد الأعلى مغلقين بين قوسين مربعين، ثم الكلمة المحجوزة «**من**» ، ثم يليها صنف خانات المنظومة.

ويتكون صنف خانات المنظومة من الأنواع التالية:

أ) الأنواع القياسية مثل الرموز الأعداد الحقيقية والأعداد الصحيحة أو منطقي.

ب) المقاييس.

ج) السلاسل الرمزية.

د)المنظومات.

و) السجلات (سوف يأتي ذكرها في الشروحات القادمة).

٩ــ٢ منظومة ذات بعد واحد (متجه)

ONE DIMENTIONAL ARRAY

هذه المنظومة هي أبسط أشكال المنظومات، وهي ذات بعد واحد حيث يمكن استدعاء أي خانة داخل المنظومة بواسطة فهرسها، أو تخصيص بعض القيم لتخزينها داخل خانات المنظومة. ويمكن تخيل هذه المنظومة على أنها صناديق بريدية مصفوفة في صف واحد أو عمود واحد، وكل صندوق معروف برقمه البريدي.

مثال:

صنف

يوم = (السبت، الأحد، الاثنين، الثلاثاء، الأربعاء، الخميس، الجمعة)؛

متغير

الاسبوع : **منظومة** [١..٧] **من** يوم؛

درجة_الاختبار : **منظومة** [١..١٠٠] **من** حقيقي؛

م : حقيقي؛

كما تعامل خانات المنظومة باستخدام فهرس كل خانة فمثلا :

أسبوع [٢] := الأحد؛

درجة_الاختبار [٧] := ٨٥,٥٠؛

م := درجة_الاختبار [٩]؛

٩ـ٣ منظومة متعددة الأبعاد (المصفوفة)

MULTI DIMINTIONAL ARRAY

بما ان المنظومة تحتوي على عدد محدود من الخانات، فإن كل خانة قد تتكون من منظومة وبذلك نسميها منظومة ذات بعدين أو مصفوفة.

وقد تتكون خانات المنظومة ذات البعدين أيضا من منظومة ولهذا نسميها منظومة متعددة الأبعاد.

مثال :

صنف

موظف = **سلسلة** [٣٠]؛

مكتب = **منظومة** [١..٢٥] **من** موظف؛

ادارة =**منظومة** [١..١٠] **من** مكتب؛

متغير

أ،ب : صحيح ؛

قسم_الحاسب_الآلي :مكتب؛

شئون_الموظفين : ادارة ؛

حيث تكون المتغيرات من نوع (ادارة) متعددة الأبعاد، كما ان حاصل

ضرب أبعاد أو حجم المنظومة يجب ان لا يخرج عن المدى (+ ٣٢٠٠٠)، فمثلا في المثال السابق يكون حجم المنظومة (ادارة) = ٣٠ × ٢٥ × ١٠ = ٧٥٠٠، كما يتم معاملة وتخصيص المنظومات على النحو التالي:

قسم_الحاسب_الآلي [ب] :=' خالد بن زيد'؛

شئون_الموظفين [٤، ٦] := 'أحمد بن عبد الله' ؛ (ه الموظف ٦ من مكتب ٤ ه)

شئون_الموظفين [٢١، ٣] := شئون_الموظفين [٢٢، ٣]؛ (ه تغيير مكتب موظف ٣ ه)

٩ـ٤ منظومة الرمز

CHARACTER ARRAY

تتكون منظومة الرمز من منظومة خاناتها مصنفة من النوع القياسي (رمز) وهي ذات فهرس واحد. وتتألف من الأحرف العربية والانجليزية وبعض الرموز الخاصة. ويمكن تخيلها مثل سلسلة رمز لها طول ثابت، ولذلك يمكن تخصيص السلسلة الرمزية الثابتة إلى منظومة الرمز اذا كانت من نفس الطول.

مثال :

صنف

اسم_ق = **منظومة** [١..٢٠] **من** رمز؛

اسم_ص = **منظومة** [١..٣٠] **من** رمز؛

ثابت

اسم_ث =' خالد بن الوليد' ؛

متغير

الاسم_الأول : اسم_ق؛

اسم_الأب، اسم_العائلة : اسم_ص؛

فيمكننا في هذه الحالة ان نخصص سلسلة رمزية ثابتة إلى منظومة من نفس الطول كما يلي :

اسم_ق := اسم_ث؛

كما يمكن تخصيص منظومة إلى أخرى اذا كانتا من صنف واحد وطول واحد مثل :

اسم_الأب := اسم_العائلة ؛

البرنامج التالي يوضع طريقة استعمال وتطبيق المنظومات حيث يقوم بترتيب منظومة تحتوي على أرقام عشوائية ترتيباً تصاعدياً.

```
برنامج ترتيب_ثنائي؛
  ثابت
    الحجم_الأعلى = ١٠٠ ؛
  صنف
    عنصر_المنظومة = منظومة [١.. الحجم_الأعلى] من صحيح؛
  متغير
    م : عنصر_المنظومة ؛
    د، ع : صحيح ؛
    {الاجراء يرتب منظومة ارقام ترتيباً تصاعديا }
  اجراء رتب (متغير م : عنصر_المنظومة؛ ع : صحيح)؛
    متغير
      وسط، اول، آخر، ف، د : صحيح؛
      مؤقت : صحيح؛
    بداية
      حيث د:= ٢  إلى ع اعمل
```

```
بداية
        مؤقت : = م [د] ؛
        آخر : = ١ ؛
        اول := د- ١؛
    {هذا التكرار يقسم المنظومة إلى قسمين ويقارن منتصفها مع اولها}
    طالما آخر < = اول اعمل
    بداية
        وسط := (آخر + اول) تقسيم ٢؛
        اذا مؤقت < م [وسط] اذن
            اول := وسط -١
        والا آخر := وسط + ١
    نهاية ؛ {طالما}
    حيث ف := د - ١ تنازلي آخر اعمل
        م [ف + ١] := م [ف]؛
        م [آخر] := مؤقت؛
نهاية؛ {حيث}
نهاية ؛ {رتب}
بداية {البرنامج الرئيسي}
    امسح_شاشة؛
    اكتب_سطر ('          ــــ طريقة الترتيب الثنائية ــــ')؛
    اكتب_سطر؛
    {كرر حتى ادخال عدد اقل من أو يساوي ١٠٠}
    كرر
        اكتب ('كم عدد العناصر (٢ إلى '، الحجم الأعلى ،')؟ ')؛
        اقرأ_سطر (ع)؛
        اكتب_سطر؛
```

```
اكتب_سطر (' الأرقام قبل ترتيبها :')؛
حتى ع < = الحجم_الأعلى؛
حيث د := ١ إلى ع اعمل
بداية
{اعمل أرقام عشوائية بعدد (ع) بداية من صفر إلى اقصى عدد صحيح}
م [د] := عشوائي (اقصى ع_ص)؛
اكتب (م [د] : ٨)؛
{اذا عدد الارقام في السطر الواحد يساوي ١٠ ابدأ من السطر التالي}
اذا (د باقي ١٠ = ٠) اذن اكتب سطر
نهاية؛
اكتب_سطر؛
رتب (م، ع)؛
{اكتب منظومة الأرقام ترتيبا تصاعديا}
اكتب_سطر (' الأرقام بعد ترتيبها : ')؛
اكتب_سطر؛
حيث د : = إلى ع اعمل
بداية
اكتب (م [د] : ٨)؛
اذا (د باقي ١٠ = ٠) اذن اكتب_سطر
نهاية؛
نهاية.
```

OOOOOOO

الباب العاشر

السجلات RECORDS

يتكون السجل من عدد محدود من البنود المختلفة التي تسمى «**حقل**» وقد يصمم كل حقل من هيكل مختلف واسم مميز عن الآخر. وتستخدم السجلات لمعالجة نوع مميز من البيانات، مثل معالجة سجلات رخص القيادة وسجلات الموظفين وعدد كبير من أنواع السجلات الأخرى.

١٠ـ١ تعريف السجلات في البرنامج

RECORDS DEFINITION

تعرف السجلات في البرنامج من الكلمة المحجوزة «**سجل**»، يتبعها قائمة تقسيم الحقول حيث الفاصلة المنقوطة (؛) تفصل كل حقل عن الآخر، وينتهي السجل بكلمة «**نهاية**» ويعرف السجل اما في منطقة التصنيف أو في منطقة المتغيرات، وتنظم بنية السجل عامة كالآتي:

اسم السجل : **سجل**

حقل ١؛

حقل ٢؛

حقل ٣؛ **حقل ٤** ؛

......

......

حقل م؛

نهاية؛

حيث يحمل اسم السجل اسم تعريف متغير مميز، كما ان الحقول يمكن ان تكون من أي من أنواع البيانات التالية:

أ) الأنواع القياسية كالرموز والأعداد الصحيحة والحقيقية ومنطقي.

ب) المقاييس.

ج) السلاسل الرمزية.

د) المنظومات.

و) السجلات.

مثال :

صنف

أيام_الشهر = ١..٣١؛

تاريخ = **سجل**

يوم : أيام_الشهر ؛

شهر: (يناير، فبراير، مارس، ابريل، مايو، يونيو، يوليو، اغسطس، سبتمبر، اكتوبر، نوفمبر، ديسمبر) ؛

سنة : ١٩٠٠..١٩٩٩؛

نهاية؛

متغير

تاريخ_الميلاد، تاريخ_التخرج، تاريخ_بدء_العمل : تاريخ ؛

يوم_الاسبوع : منظومة [١..٧] **من** تاريخ؛

سنة : حقيقي ؛

المتغيرات (يوم) و (شهر) و (سنة) عبارة عن حقول للسجل (تاريخ)، حيث

كل واحد منها مميز عن الآخر، ونقصد بمميز ..! أي كل حقل يحتوي على اسم متغير يختلف عن الحقل الآخر، ويشترط فقط داخل السجل، أما خارجه فنستطيع استخدام نفس اسم الحقل لنعرفه تعريفا آخراً كما في المثال السابق، حيث عرفنا المتغير (سنة) مرتين : الأولى داخل السجل كمقياس والأخرى في منطقة المتغيرات كعدد حقيقي. وتتم معاملة الحقول وتعيينها وادخال المعلومات فيها كالآتي:

تاريخ_الميلاد.شهر: = اكتوبر؛

تاريخ_التخرج.سنة:= ١٩٨٠؛

يوم_الاسبوع [٤].أيام الشهر := ٧؛ (* تاريخ اليوم الرابع من الأسبوع = ٧ *)

وتتم معاملة السجلات مع بعضها كمعاملة المنظومات مع بعضها البعض، حيث يمكن تعيين سجل إلى آخر اذا كانا متطابقين.. أي معرفين من نفس نوع السجل مثل:

تاريخ_بدء_العمل := تاريخ_التخرج؛

١٠ــ٢ العبارة (مع)

WITH STATEMENT

لقد عرفنا من المثال السابق عند ادخال البيانات إلى حقول السجل انه يجب كتابة اسم السجل ثم الحقل في كل مرة، لكن هذه الطريقة تعتبر شاقة وخاصة عندما نريد ادخال بيانات كثيرة، الا اننا نستطيع اختصار هذه الطريقة باستخدام العبارة (مع)، حيث نذكر اسم السجل المراد استخدامه مرة واحدة فقط حسب الصيغة (**مع سجل .. اعمل**)، كما يمكن استخدام التعبير (مع) في أكثر من سجل دفعة واحدة كالآتي:

مع سجل أ، سجل ب، سجل ج، ....، سجل ي **اعمل**

أما اذا كانت بعض السجلات مشتركة باسم واحد في حقولها، فاننا نستطيع ان نرتب الترتيب السابق بطريقة أخرى لكي لا يجعل أي التباس كالآتي:

مع سجل أ **اعمل**

مع سجل ب **اعمل**

مع سجل ج **اعمل**

مع سجل ي **اعمل**

```
برنامج عبارة_مع؛
صنف
        سنة = (اولى، ثانية، ثالثة، رابعة)؛
        جدول = سجل
            رقم_الطالب : ١٠٠..٩٩٩٩ ؛
            المعدل_التراكمي : رمز ؛
            مجموع_الدرجات : حقيقي ؛
            السنة_الاكاديمية : سنة ؛
        نهاية ؛
        العنوان = سجل
            ص_ب : صحيح
            اسم_الشارع : سلسلة [٢٥]؛
            المدينة : سلسلة [٢٠]؛
            رمز_بريدي : صحيح
        نهاية؛
        ملف_الطالب = سجل
            اسمه : سلسلة [٢٠] ؛
            عنوانه : العنوان ؛
        نهاية
        تقرير_الطالب = سجل
```

```
                    ملفه : ملف_الطالب؛
                    تقريره : جدول؛
                    نهاية ؛
متغير
          التقرير : منظومة [١..٩٩] من تقرير_الطالب؛
          اسم : سلسلة [٣٠]؛
          م، ل : صحيح؛
بداية
   مع التقرير [٩٨] اعمل
       مع ملف اعمل
          بداية
             اسمه :=' احمد فهد عبد الرحمن'؛
             مع عنوانه اعمل
                بداية
                   ص_ب := ٦٧٨٢؛
                   اسم_الشارع := 'خالد بن الوليد'؛
                   المدينة := 'الرياض'؛
                   رمز_بريدي := ١٢٣٧٦؛
                نهاية ؛
             مع تقريره اعمل
                بداية
                   رقم_الطالب := ٩٨؛     المعدل_التراكمي := 'ب' ؛
                   مجموع_الدرجات := ١٢٤٣,٢٥؛     السنة_الاكاديمية := ثالثة ؛
                نهاية ؛
          نهاية ؛
نهاية .
```

١٠ـ٣ السجلات ذات الحقول المتغيرة

RECORDS WITH VARIABLE FIELDS

لاحظنا في الأمثلة السابقة ان حقول السجلات ثابتة لا تتغير في السجل، ولذلك فإن تعريف جملة السجل يسمح بان نعرف حقول متغيرة اذا دعت الحاجة اليها بحيث يأخذ كل حقل عدة اختيارات من الحقول الفرعية، ويتم تعريف هذه الحقول المتغيرة باستخدام العبارة (**حالة**).

مثال :

صنف

عنصر = (عربي، أجنبي)؛

البطاقة_الشخصية = **سجل**

الاسم : **سلسلة** [٣٠] ؛

العنوان : **منظومة** [١..٥٠] **من** رمز ؛

حالة الأصل : عنصر **من**

عربي : (رقم_البطاقة : حقيقي) ؛

أجنبي : (الجنسية : **سلسلة** [٢٠] ؛

رقم_الجواز : حقيقي)؛

نهاية؛

يجب ملاحظة أن تعريف الحقل المتغير يأتي بعد تعريف جميع الحقول الثابتة، وكل سجل لا يجب احتوائه على أكثر من حقل متغير واحد فقط، الا أن واحدا من حقول الحقل المتغير قد يكون متغير أيضاً، ويجب وجود

القوسين () حتى اذا كان الحقل الفرعي فارغ.

البرنامج التالي يوضح طريقة تطبيق السجلات في البرامج حيث يعمل سجلات خاصة بأعضاء نادي معين ثم يبين بعض الطرق لتداولها.

```
برنامج بيانات_النادي ؛
ثابت
    اقصى_اعضاء = ٥ ؛
صنف
    عضو_النادي = سجل
        اسم : سلسلة [٣٠]؛
        تلفون : سلسلة [١٢] ؛
        حساب : حقيقي ؛
        قسط : حقيقي ؛
    نهاية ؛
متغير
    اعضاء      : منظومة [١.. اقصى_اعضاء] من عضو_النادي ؛
    عضوية      : ٠.. اقصى_اعضاء ؛
    اختيار ، ح : رمز ؛
(*          يضيف الاجراء سجلا إلى قائمة الاعضاء *)
اجراء اضافة_سجل ؛
بداية
    اذا عضوية = اقصى_اعضاء
    اذن اكتب_سطر (' العضوية مليئة') والا
بداية
    اكتب_سطر (' اضف سجل جديد') ؛
    عضوية : = عضوية + ١ ؛
مع اعضاء [عضوية] اعمل
```

```
بداية
        اكتب (' الاسم          :') ؛ اقرأ_سطر (اسم)؛
        اكتب (' تلفون          :') ؛ اقرأ_سطر (تلفون)؛
        اكتب (' الحساب         :') ؛ اقرأ_سطر (حساب)؛
        اكتب (' القسط          :') ؛ اقرأ_سطر (قسط) ؛
    نهاية (• مع •)
  نهاية (• والا •)
نهاية ؛ (• اضافة سجل •)
(•    يعطي الاجراء قائمة سجلات أعضاء النادي         •)
 اجراء قائمة_سجل؛
    متغير
        رقم : صحيح ؛
  بداية
    اكتب_سطر (' قائمة الاعضاء :') ؛
    حيث رقم := ١ إلى عضوية اعمل
    مع اعضاء [رقم] اعمل
        اكتب_سطر (رقم، ' : '، اسم، ' : '،' _ '، تلفون، '_' ،
                  حساب - قسط : ٤ : ٢)؛
  نهاية ؛ (• قائمة سجل •)
بداية (• البرنامج الرئيسي •)
    امسح_شاشة ؛
    اكتب_سطر (' •••         بيانات النادي       ••') ؛
    عضوية : = ٠؛
    كرر
      اكتب_سطر ؛
      اكتب_سطر (' عدد الاعضاء = '، عضوية)؛
      اكتب_سطر ؛
```

```
اكتب (' (أ) ضف سجل، (ق)ئمة السجلات، (ن)هاية؟') ؛
اقرأ_سطر (اختيار)؛
حالة اختيار من
    'أ' : اضافة_سجل
    'ن' : اكتب_سطر ('انتهى') ؛
    'ق' : قائمة_سجل
    والا اكتب_سطر (' الرمز غير واضح! كرر الادخال')؛
  نهاية (* حالة *)؛
  اذهب_ع_س (٣٠، ٢٣)؛
  اكتب (' اضغط أي مفتاح للعودة') ؛
  اقرأ (ل_م، ح) ؛
  امسح_شاشة ؛
حتى (اختيار = 'ن') ؛
  امسح_شاشة ؛
نهاية.
```

OOOOOOO

الباب الحادي عشر

المجموعات SETS

تعتبر المجموعات احدى الفروع الأساسية في الرياضيات، فلها عملياتها وخواصها الخاصة بها وقد تم تطبيق بعض هذه العمليات والخواص في بعض لغات برمجة الحاسب الآلي كوسيلة لايجاد حلول لبعض العمليات التي تتطلب استعمال المجموعات. وسميت المجموعة بهذا الاسم لأنها تحتوي على أجزاء مرتبطة ببعضها البعض، وكل جزء يسمى عضو أو عنصر من هذه المجموعة، ففي كثير من الاحيان يريد المبرمج أن يعبر عن مجموعة من العناصر مثل :

أ) الأحرف الهجائية.

ب) الأعداد المفردة من ١ إلى ٩٩.

ج) الألوان.

ج) أشهر السنة.

والكثير من هذه المجموعات التي لا يمكن حصرها.

١١ــ١ تعريف المجموعة في البرنامج

SETS DEFINITION

هي عبارة عن مجموعة من عناصر لا يتعدى عددها ٢٥٦ عنصراً وتكون من نوع واحد، بسيط كالمقاييس، والرموز والاعداد الصحيحة .. وان لا تكون من نوع (حقيقي)، ويتم تعريف المجموعة باستخدام الكلمة المحجوزة «**مجموعة**» يتبعها نوع بسيط، ويسمى نوع العناصر **بالنوع الأساسي**.

أمثلة :

صنف

الأرقام = **مجموعة من** ٠..٩ ؛

الرموز = **مجموعة من** رمز ؛

ألوان_الورد = **مجموعة من** (أحمر، أبيض، أصفر، بنفسجي)؛

الأحرف_الانجليزية = **مجموعة من** 'A'.. 'Z'؛

المقاييس مشابهة للمجموعات تقريباً، حيث الحد الأعلى لعدد عناصر المجموعة **٢٥٦** عنصر، كما يجب أن تكون عناصر مجموعة الأعداد الصحيحة داخل المدى (**٠..٢٢٥**)، وتتميز المجموعات بوجود عمليات بينها سيأتي ذكرها في الشروحات القادمة.

١١ــ٢ انشاء المجموعات

SET CONSTRUCTING

يمكن للمجموعة أن تحوي ٢٥٦ عنصرا فأقل من عناصر مجموعة النوع الأساسي وتوضع المجموعة بين قوسين مربعين، وتفصل الفاصلة بين عناصرها، أو يوضع الحد الادنى والحد الأعلى منها، وتحمل جميع خصائص مجموعة النوع الأساسي.

أمثلة :

[١..٦٠] مجموعة نوعها الأساسي الأعداد الصحيحة.

['أ'، 'ب'، 'ج'، 'د'] مجموعة نوعها الأساسي الرموز.

[] المجموعة الفارغة.

المجموعة الفارغة في المثال الأخير تنتمي إلى كل أنواع المجموعات، ويجب ملاحظة أن التعبير [١..٦] يقصد به المجموعة [١، ٢، ٣، ٤، ٥، ٦]، وتفيد المجموعات بأنها تختصر الكثير من التعبيرات المطولة مثل :

اذا (ح = 'أ') **أو** (ح = 'ج') **أو** (ح = 'د') **اذن**

نسنطيع كتابتها كالآتي:

اذا ح **في** ['أ'، 'ج'، 'د'] **اذن**

والتعبيرات متساويان في المعنى ويؤديان نفس الغرض.

١١ــ٣ عمليات المجموعات

OPERATIONS ON SETS

للمجموعات خصائص وعوامل مميزة، حيث أننا نستطيع أن نقول على مجموعتين بأنهما متساويتان اذا كانت عناصرهما متساوية، كما ان ترتيب العناصر غير معتبر لحصول التساوي، ويجب عند اجراء هذه العمليات بين المجموعات أن تكون من نوع واحد، كما ان تقييم هذه العمليات يأتي حسب أفضلية معينة اذا وجد أكثر من عملية واحدة في التعبير وترتيب هذه الأفضلية مدرج حسب شرحنا لهذه العمليات ومن أهمها:

١) **اتحاد المجموعات** : فحاصل اتحاد أو جمع المجموعة ١ مع المجموعة ٢ هو المجموعة التي تحتوي على جميع عناصر المجموعة ١ مع عناصر المجموعة ٢ التي لا توجد في المجموعة ١، ويعبر عنها بالأداة (+).

مثال :

مجموعة_١ = [٢، ٤، ٦، ٨، ١٠]

مجموعة_٢ = [١، ٢، ٣، ٤]

مجموعة_١ + مجموعة ٢ = [١، ٢، ٣، ٤، ٦، ٨، ١٠]

٢) **تقاطع المجموعات** : فنتيجة تقاطع المجموعة ١ مع المجموعة ٢ هو المجموعة التي تحتوي على عناصر المجموعة ٢ وتوجد في المجموعة ١، ويعبر عن العملية بالأداة (**x**).

مثال :

باعتبار نفس قيم المجموعة ١ والمجموعة ٢ في المثال السابق فإن

مجموعة ١ **x** مجموعة ٢ = [٢، ٤]

٣) **اتمام المجموعات** : فنتيجة اتمام المجموعة ٢ من المجموعة ١ هو المجموعة التي تحتوي على عناصر المجموعة ١ والتي ليست في المجموعة ٢، ويعبر عن العملية بالأداة (–).

مثال :

باعتبار نفس القيم في المثالين السابقين للمجموعتين فإن

مجموعة_١ ـــ مجموعة_٢ = [٦، ٨، ١٠]

٤) **مقارنة المجموعات** : وهي مقارنات منطقية.. أي يكون ناتج المقارنة اما «**صح**» أو «**خطأ**»، وتتم مقارنة المجموعات باستخدام عوامل المقارنة المعروفة وهي :

= : مقارنة التساوي بين مجموعتين .. أي احتوائها على نفس العناصر.

< >: مقارنة عدم التساوي بين مجموعتين.. أي احتواء كل منهما على عناصر مختلفة.

‎<= :‎ تعتبر نتيجة المقارنة «صح» اذا كانت جميع عناصر الطرف الأيسر موجودة في الطرف الأيمن من العملية.

‎>= :‎ تعتبر نتيجة المقارنة «صح» اذا كانت جميع عناصر الطرف الأيمن موجودة في الطرف الأيسر من العملية.

في : وتستخدم لاختبار العضوية، حيث أن الطرف الأيسر يكون مجموعة والطرف الأيمن يكون عنصرا من نفس نوع عناصر المجموعة، وتكون نتيجة الاختبار «**صح**» اذا كان العنصر موجودا في المجموعة والا «**خطأ**» اذا لم يوجد.

١١ـ٤ تخصيص المجموعات

SETS ASSIGNMENTS

يستخدم عامل التخصيص (:=) لتخصيص أو تعيين مجموعة أو نتيجة عملية من عمليات المجموعات إلى متغير، ويسمى هذا المتغير بمتغير المجموعة، ويكون معرفا إلى نفس نوع الأساس الذي تنتمي له المجموعة المخصصة له.

مثال :

صنف

مواد_دراسية = **مجموعة من** (رياضيات، فيزياء، كيمياء، أحياء، جيولوجيا، ميكانيكا، نصوص، قواعد، جغرافيا، تاريخ، نقد وبلاغة، انجليزي، تربية رياضية)؛

متغير

مواد_علمية، مواد_أدبية، مواد_مشتركة : مواد_دراسية؛

```
بداية
    مواد_علمية := [رياضيات.. ميكانيكا، انجليزي، تربية رياضية]؛
    مواد_أدبية := [نصوص.. تربية رياضية]؛
    مواد_مشتركة := مواد_علمية × مواد_ادبية؛
نهاية؛
```

البرنامج الآتي يوضح طريقة استخدام المجموعات في البرامج وتطبيق بعض العمليات عليها.

```
برنامج عمليات_المجموعة؛
  صنف
      اشهر = (يناير، فبراير، مارس، ابريل، مايو، يونيو،
          يوليو، اغسطس، سبتمبر، اكتوبر، نوفمبر، ديسمبر)
      مجموعة_اشهر = مجموعة من اشهر؛
  متغير
      اسماء_الاشهر : منظومة [اشهر] من سلسلة [٦]،
      مجموعة_١، مجموعة_٢ : من اشهر؛
      {الاجراء يعطي قائمة بالمجموعة المعطاة}
      اجراء اظهر_المجموعة (م: مجموعة_اشهر)؛
        متغير
            فاصل : سلسلة [٢]؛
            شهر_واحد : اشهر؛
        بداية
            اكتب ('[') ؛
            فاصل := ' '،
            حيث شهر_واحد := يناير إلى ديسمبر اعمل
            اذا شهر_واحد في م اذن
```

```
بداية
    اكتب (فاصل، اسماء الاشهر [شهر واحد]؛
    فاصل:= '،'
        نهاية؛ {اذا}
        اكتبسطر (']')؛
      نهاية، {اظهر المجموعة}
بداية
  اسماء_الاشهر [يناير] := 'يناير'؛          اسماء_الاشهر [فبراير] := 'فبراير'؛
  اسماء_الاشهر [مارس] := 'مارس'؛            اسماء_الاشهر [ابريل] := 'ابريل'؛
  اسماء_الاشهر [مايو] := 'مايو'؛            اسماء_الاشهر ['يونيو] := 'يونيو'؛
  اسماء_الاشهر [يوليو] := 'يوليو'؛          اسماء_الاشهر [اغسطس] := 'اغسطس'؛
  اسماء_الاشهر [سبتمبر] := 'سبتمبر'؛        اسماء_الاشهر [اكتوبر] := 'اكتوبر'؛
  اسماء_الاشهر [نوفمبر] := 'نوفمبر'؛        اسماء_الاشهر [ديسمبر] := '؛
  مجموعة_١ := [يناير، فبراير، مارس]؛
  مجموعة_٢ := [فبراير، مارس، ابريل]؛
  امسح_شاشة؛
  اكتبسطر ('          عمليات المجموعات')؛
  اكتبسطر ('          ____________')؛
  اكتبسطر(' المجموعة الاولى =' )؛ اظهر_المجموعة (مجموعة_١)؛
  اكتبسطر( المجموعة الثانية =')؛ اظهر_المجموعة (مجموعة_٢)؛
  اكتبسطر (' (اتحاد) المجموعة_الاولى + المجموعة_الثانية =')؛
  اظهر_المجموعة (مجموعة_١ + مجموعة__٢)؛
  اكتبسطر (' (تقاطع المجموعة الاولى × المجموعة الثانية =')؛
  اظهر_المجموعة (مجموعة_١ × مجموعة_٢)؛
  اكتب (' (فرق) المجموعة الاولى - المجموعة الثانية =')؛
  اظهر_المجموعة (مجموعة_١ - مجموعة_٢)؛
نهاية .
```

الباب الثاني عشر

الثوابت CONSTANTS

تمهيد

لا يخصص البرنامج قيم ابتدائية للمتغيرات عند البداية وانما هذا من عمل المبرمج، وقد يحتاج المبرمج إلى استعمال قيمة معينة بكثرة داخل البرنامج، فمثلا لحساب قيمة الأرباح المركبة المكونة من ٦٪ في حسابات بنك، مما يتطلب كتابة هذا الرقم مئة مرة في البرنامج، فإذا ارتفعت هذه الارباح إلى ٨٪ وأردنا تغييرها في البرنامج سنحتاج إلى تغييرها مئة مرة، ولكن اذا كان هناك متغير يسمى (نسبة_الربح) مثلا وأعطى هذه القيمة فإن المبرمج لا يحتاج الا لتغيير قيمة هذا المتغير فقط، ولذلك صمم في البرنامج منطقة لتعريف هذا النوع من المتغيرات وهي **منطقة الثوابت**. الثوابت عبارة عن متغيرات تخصص لها قيم ابتدائية عند بداية البرنامج، حيث تكون هذه القيم مستخدمة بكثرة، وتكون القيم من أنواع المتغيرات البسيطة المعروفة لدينا، كما ان هذه القيم تبقى ثابتة .

ويوجد نوعان من الثوابت : **الثوابت المصنفة والغير مصنفة**، فالغير مصنفة تبقى ثابتة في البرنامج ولا يمكن تغيير قيمتها داخله، ولكن في الثوابت المصنفة يمكن ذلك، والفرق الوحيد في تعريفهما هو إضافة صنف المتغير الثابت في الثوابت المصنفة، حيث يتبع المتغير النقطتين (:) يليها نوع الثابت المصنف ثم علامة التساوي (=) ثم القيمة الثابتة.

١٢ـ١ الثوابت الغير مصنفة

UNTYPED CONSTANTS

تعرف الثوابت المصنفة في منطقة تعريف الثوابت وذلك بتخصيص قيمة مبدئية من أحد الأصناف القياسية إلى متغير، ولا يمكن تخصيص أي قيمة أخرى لهذا المتغير بعد ذلك داخل البرنامج، ويمكن تمرير هذا النوع كمعطيات اجراء ولكن ليس بطريقة الاسناد وانما بطريقة القيم.

أمثلة :

ثابت

س = 'أهلا وسهلا' ؛

نسبة الربح = ٠,٠٦ ؛

ع = ١٠٠؛

١٢ـ٢ الثوابت المصنفة البسيطة

SIMPLE TYPED CONSTANTS

يمكن تعريف المقاييس القياسية إلى هذا النوع من الثوابت.

أمثلة :

ثابت

عدد الطلبة : صحيح = ٣٥٨٦ ؛

نسبة النجاح : حقيقي = ٩١,٨٥ ؛

الامتحان : **سلسلة** [٧] = 'رياضيات' ؛

التقدير : رمز = ب ؛

يجب ملاحظة أن الثوبت المصنفة تأتي على عكس الثوابت العادية من حيث انها قد تأتي في معطيات الدالة أو الاجراء بطريقة الاسناد وطريقة القيم،

ولا تأتي في أي مكان آخر لتعريف المتغيرات، كما يمكن تخصيصها إلى قيم أخرى في البرنامج.

مثال :

ثابت

أدنى : صحيح = ٠ ؛

أعلى : صحيح = ١٠٠ ؛

صنف

درجة_الاختبار : **منظومة** [أدنى .. أعلى] **من** حقيقي ؛

فتعريف (درجة_الاختبار) في المثال السابق غير مسموح به اطلاقا.

١٢ــ٣ الثوابت المصنفة الهيكلية

STRUCTURED TYPED CONSTANTS

تصاغ الثوابت المصنفة الهيكلية من المنظومات الثابتة والسجلات الثابتة والمجموعات الثابتة. وتستخدم لإعطاء قيم مبدئية لقوائم ومجموعات، لتستخدم في المقارنة أو التحويل.

١٢ــ٣ــ١ المنظومات الثابتة

CONSTANT ARRAYS

المنظومات الثابتة نوعان كما نعرفها في المنظومات وهي المنظومات ذات البعد الواحد والمنظومات المتعددة الأبعاد.

ويشمل تعريف **المنظومات الثابتة ذات البعد الواحد** من متغير يتبعه الأداة (:) ثم يأتي متغير منظومة يكون مصنفا من قبل، ثم اداة التساوي (=) ، يليها قيم المنظومة مسلسلة كمجموعة ثوابت.

مثال :

صنف

لون = (أحمر، أصفر، أخضر)؛

سلسلة_لون = **منظومة** [لون] **من سلسلة** [٤]؛

ثابت

اشارة : سلسلة_لون = ('أحمر'، 'أصفر'، 'أخضر')؛

فالمثال يعرف المنظومة الثابتة (اشارة) ، والتي تستخدم لتحويل قيم المقياس إلى قيمها المطابقة من نوع سلسلة، وتكون كالاتي:

اشارة [أحمر] = 'أحمر'

اشارة [أصفر] = 'أصفر'

اشارة [أخضر] = 'أخضر'

وقد تكون قيم المنظومة من أي نوع من الأنواع القياسية المعروفة، الا من نوع «ملف» أو نوع «مؤشر»، كما يمكن تعريف منظومة الرمز كرمز واحد أو سلسلة.

مثال :

ثابت

أرقام : **منظومة** [٠..٩] **من رمز** =

('٠'، '١'، '٢'، '٣'، '٤'، '٦'، '٧'، '٨'، '٩')؛

كما يمكن صياغتها كسلسلة رموز كالآتي:

أرقام : **منظومة** [٠..٩] **من رمز** = '٩٨٧٦٥٤٣٢١'؛

وتعرف **المنظومات الثابتة المتعددة الأبعاد** في البرنامج بوضع قيم كل بعد في مجموعة مستقلة عن الأخرى بين قوسين، ويفصل بين المجموعات الفاصلة (،) ، فتكون القيم الداخلية مخصصة للأبعاد اليسرى.

```
برنامج متعدد_أبعاد؛
صنف
مربع = منظومة [٠.. ١، ٠.. ١، ٠.. ١] من صحيح؛
ثابت
البعد : مربع = (((٠،١)، (٢،٣))، ((٤،٥)، (٦،٧)))؛
بداية
    اكتب_سطر (البعد [٠،٠،٠]، ' = ٠')؛
    اكتب_سطر (البعد [٠،٠،١]، '= ١')؛
    اكتب_سطر (البعد [٠،١،٠]، ' = ٢')؛
    اكتب_سطر (البعد[٠،١،١]، ' = ٣')؛
    اكتب_سطر (البعد[١،٠،٠]، ' = ٤')؛
    اكتب_سطر (البعد[١،٠،١]، ' = ٥')؛
    اكتب_سطر (البعد[١،١،٠]، ' = ٦')؛
    اكتب_سطر (البعد[١،١،١]،' =' ٧')؛
نهاية.
```

١٢ــ٣ــ٢ السجلات الثابتة

CONSTANT RECORDS

تصاغ السجلات الثابتة بكتابة اسم متغير ثم علامة (:) ، يليها اسم سجل معرف من قبل ، ثم اشارة التساوي (=)، ثم قيم حقول السجل الثابتة موضوعة في قائمة مفصولة بالفاصلة المنقوطة (؛) ومغلقة بين قوسين.

```
مثال :
صنف
    نوع_التأشيرة = (زيارة، سياحة، عمل)؛
    بطاقة_شخصية = سجل
        الأسم : سلسلة [٣٠]؛
```

الرقم : صحيح؛
تأشيرة : نوع_التأشيرة ؛
نهاية؛

ثابت

القائمة_السوداء : **منظومة** [١..٣] **من** بطاقة_شخصية = (((الاسم : ' الوردة الحمراء')؛
(الرقم : ٨٧٦٥)؛
(تأشيرة : سياحة))،
((الاسم : ' الوددردة البيضاء')؛
(الرقم : ٥٧٦٠)؛
(تأشيرة : زيارة))،
((الاسم : 'الورده الصفراء')؛
(الرقم : ٨٩٧٨)؛
(تأشيرة : عمل)))؛

يلاحظ أن ترتيب الحقول الثابتة هو نفس ترتيبها في تعريف سجلها، وإذا كانت حقول السجل من نوع «ملف» أو من نوع «مؤشر» فإن هذا السجل لا يمكن تعريف حقوله كحقول ثابتة.

١٢ـ٣ـ٣ المجموعات الثابتة

CONSTANT SETS

المجموعات الثابتة عبارة عن مجموعة تحتوي على عنصر أو عضو فأكثر، وقد يتكون العنصر من مدى معين، حيث يكتب الحد الأدنى والأعلى من المدى مفصولين بنقطتين بينهما، وتكون هذه العناصر مفصولة فيما بينها بالفاصلة (،) وموضوعة جميعها بين قوسين مربعين.

مثال :

صنف

رموز عربية = **مجموعة من** 'أ'.. 'ي'؛

ثابت

الأرقام : رموز عربية = ['٠'..'٩']؛

المحددات : **مجموعة من** رمز = [' '..')'،'،'/'..'؛'،':'،'<'..'>' ،'

'['..']'،''{'..'}'[؛

○○○○○○○

لا يتم تحديده وإنما يتحدد بسعة أداة التخزين التي يخزن عليها، وسوف نوضح في هذا الجزء بعض أنواع الملفات المستعملة في نظام لغة نبراس.

١٣ـ١ تعريف الملف في البرنامج

FILE TYPE DEFINITION

يتكون الملف من مجموعة من الفقرات حيث كل فقرة تكون مصنفة من الأصناف القياسية المعروفة (حقيقي، صحيح، رمز، سلسلة، منظومة، سجل)، ويعرف الملف في البرنامج باستخدام الكلمتين المحجوزتين «**ملف من**» يتبعها نوع الملف، فيكون اما نوعا بسيطا أو نوعا مركبا، إلا ان النوع لا يمكن ان يكون من نوع «ملف».. أي ان مكونات الملف لا تتكون من ملفا آخر. ويأتي تعريف الملف اما كنوع في منطقة «صنف» أو كمتغير في منطقة «متغير» ويسمى هذا النوع من الملفات «**الملفات المصنفة**» ، وتنظم فقراتها على هيئة سجلات حيث كل فقرة تتكون من سجل، ويتعين عدد وحدات تخزين السجل (البايت) بعدد الوحدات اللازمة لنوع بيانات السجل، فمثلا اذا كان السجل من صنف (حقيقي).. فمعنى ذلك أن كل سجل في الملف سيحتجز ست وحدات ذاكرة.

مثال :

صنف

استيراد = **سجل**

البصاعة : **سلسلة** [٣٥]؛

عدد : صحيح؛

نهاية؛

نوع_العمل = (عامل، موظف، اداري، فني، مهندس، مدير_ادارة)؛

موظف = ملف من سجل

اسم_الموظف : **سلسلة** [٣٠]؛

الوظيفة : نوع_العمل؛

نهاية؛

متغير

طلب_استيراد : **ملف من** استيراد؛

موظفين : موظف؛

١٣ــ٢ عمليات الملفات

OPERATIONS ON FILES

في هذه الفقرة سوف نوضح العمليات المتداولة في الملفات، وكيفية استعمالها فتكون موزعة على اجراءات ودوال قياسية، وكل اجراء له وظيفة معينة.

١٣ــ٢ــ١ الاجراءات القياسية على الملفات

STANDARD PROCEDURES ON FILES

الاجراءات القياسية التالية تستخدم في عمليات الملفات :

خصص

تركيب الجملة : خصص (اسم_ملف، س)؛

يخصص هذا الاجراء سلسلة الرموز الموجودة في تعبير السلسلة (س) إلى (اسم_ملف) وهو اسم متغير لملف معرّف، وتعرف (س) كسلسلة من طول معين، فيخزن فيها اسم ملف يكون مخزنا في اسطوانة ويراد تداوله في البرنامج ثم يخصص إلى متغير ملف حيث لا يستطيع البرنامج اجراء العمليات الا على متغير ملف، كما ان جميع العمليات المتداولة على هذا المتغير سوف يقوم

البرنامج باجرائها على الملف المخزن.

افتح_م_ك

تركيب الجملة : افتح_م_ك (اسم_ملف)؛

يقوم هذا الاجراء بصنع ملف جديد فارغ في اسطوانة التخزين ويحمل اسم المتغير المخصص (اسم_ملف)، كما يعده للتشغيل والكتابة، حيث يقوم بوضع مؤشر الملف إلى بداية الكتلة الأولى منه، كما أنه اذا كان هناك ملف موجود من قبل بنفس الاسم، فسوف يمسحه نظام اللغة ويضع مكانه الملف الجديد.

افتح_م_ق

تركيب الجملة : افتح_م_ق (اسم_ملف)؛

يقوم هذا الاجراء باعداد وتجهيز ملف موجود في أداة التخزين ومخصص للمتغير (اسم_ملف) للتشغيل والقراءة، حيث يقوم بوضع مؤشر الملف في بداية الملف، كما انه يجب وجود هذا الملف في أداة التخزين والا ستحدث أخطاء في عملية الاخراج والادخال.

أقرأ

تركيب الجملة : أقرأ (اسم_ملف، م١، م٢، ...)؛

يقرأ هذا الاجراء متغير (م١) أو أكثر من متغيرات الملف المفتوح واحدا بعد الآخر من اسطوانة التخزين، ويقوم مؤشر الملف بالاتجاه إلى الفقرة التالية بعد قراءة كل متغير، ويكون الملف (اسم_ملف) قد خصص له ملف على اسطوانة التخزين من قبل بواسطة الاجراء «خصص».

اكتب

تركيب الجملة : اكتب (اسم_ملف، م١، م٢، ..)؛

يكتب الاجراء متغير (م١) أو أكثر من متغيرات الملف المفتوح واحدا بعد الآخر إلى اسطوانة التخزين، ويقوم مؤشر الملف بالاتجاه إلى الفقرة التي تليها بعد كتابة كل متغير، ويكون الملف (اسم ملف) قد خصص له ملف على اسطوانة التخزين من قبل بواسطة الاجراء «خصص».

بحث_عن

تركيب الجملة : بحث_عن (اسم_ملف، ن)؛

يتحكم هذا الاجراء بمؤشر الملف ويقوم بتوجيهه إلى الفقرة (ن)، حيث (ن) عبارة عن عدد صحيح يمثل رقم فهرس الفقرة، كما ان رقم الفقرة الأولى في الملف دائما تبدأ بصفر ويفيد هذا الاجراء عند ارادة تمديد طول الملف أو الكتابة في نهايته، ويمكن تعيين الفقرة التي قبل الأخيرة باستخدام التعبير:

بحث_عن (اسم_ملف، **حجم_ملف** (اسم_ملف))؛

فهذا التعبير يضع مؤشر الملف في نهاية الملف، فالدالة «حجم_ملف» تقوم بارجاع عدد الفقرات الموجودة فيه، وهذا العدد يكون أكبر بواحد من رقم فهرس آخر فقرة الملف حيث أن رقم الفهرس يبدأ من الصفر.

ارسل_إلى_ملف

تركيب الجملة : ارسل_الى_ملف (اسم_ملف)؛

يعمل هذا الاجراء على افراغ الذاكرة الوسيطة في قطاع الاسطوانة والمخصصة لعمليات الملف (اسم_ملف)، ويعمل فقط على ملفات النص، ولا يستعمل على الملفات المغلقة.

اغلق

تركيب الجملة : اغلق (اسم_ملف)؛

يعمل على اغلاق الملف المخزن في اسطوانة التخزين والمرتبط اسمه بالمتغير (اسم_ملف)، كما يقوم بتجديد معلومات دليل الملفات بالغاء المعلومات القديمة ثم وضع المعلومات الجديدة في مكانها كما يجب اغلاق أي ملف بعد استعمال الاجراء «افتح_م_ك» أو الاجراء «افتح_م_ق» قبل الخروج من البرنامج.

امسح

تركيب الجملة : امسح (اسم_ملف)؛

يمسح الملف المخزن في اسطوانة التخزين والمرتبط اسمه بالمتغير (اسم_ملف)، واذا استخدم هذا الاجراء بعد الاجراء «افتح_م_ق» أو الاجراء «افتح م ك»، فمن الأفضل اغلاق الملف بالإجراء «اغلق» قبل مسح الملف.

غير_اسم

تركيب الجملة : غير_اسم (اسم_ملف، س)؛

يعمل هذا الاجراء على تغيير اسم الملف المرتبط بالمتغير (اسم_ملف) ويعطيه الاسم الجديد الموضوع في السلسلة (س)، كما يقوم بتغيير دليل الملفات ووضع هذا الاسم فيه ايضا، وتتم بعد ذلك جميع العمليات على الملف باستخدام الاسم الجديد، ويلاحظ أن هذا الاجراء لا ينفذ والملف مفتوح، ويجب على المبرمج أيضا أن لا يستخدم اسما معرفا في دليل الملفات.

مثال :

صنف

اسم = سلسلة [٣٥] ؛

.

.

.

دالة هل_موجود (اسم_ملف : اسم) : منطقي؛

متغير

م : **ملف؛**

بداية

خصص (م، اسم_ملف)؛

{$م+} افتح_م_ق (م)؛ {$م-}

موجود := (نتيجة_دخ =٠)

نهاية؛

في المثال السابق ، الدالة «هل_موجود» تعطي نتيجة منطقية.. وهي صحيحة اذا كان الملف موجودا في دليل التخزين، وخطأ اذا لم يكن موجودا أو الاسم غير معروف، والدالة «نتيجة_د_خ» تعطي نتيجة عدد صحيح، وتعمل على فحص عملية الادخال والاخراج في الملف فإذا كانت النتيجة صفر فالعملية صحيحة أما اذا كانت غير ذلك فيعني حدوث خطأ في آخر عملية ادخال أو اخراج من الملف.

برنامج تغيير؛

متغير

اسم_ملف : **ملف؛**

اسم : سلسلة [٢٠]؛

بداية

اكتب (' ادخل الأسم القديم للملف : ')؛ اقرأ_سطر (اسم)؛

خصص (اسم ملف، اسم)؛

اكتب ('ادخل الأسم الجديد للملف :')؛ اقرأ_سطر (اسم)؛

غير_اسم (اسم_ملف، اسم)؛

نهاية.

١٣ـ٢ـ٢ الدوال القياسية على الملفات

STANDARD FUNCTIONS ON FILES

الدوال التالية عبارة عن دوال قياسية ضرورية لعمليات الملفات:

ن_م

تركيب الجملة : ن_م (اسم_ملف)؛

ترمز تركيب الدالة «ن_م» لكلمة نهاية ملف، ونتيجتها من نوع منطقي، وهي صحيحة اذا كان مؤشر الملف موضوع في نهاية الملف، وخطأ اذا كان المؤشر في أي مكان آخر.

موضع_ملف

تركيب الجملة : موضع ملف (اسم_ملف)؛

نتيجة الدالة من نوع عدد صحيح، ويمثل هذا العدد موضع المؤشر في الملف، فإذا كان العدد صفر فإن موضع المؤشر في بداية الملف.

حجم_ملف

تركيب الجملة : حجم_ملف (اسم ملف)؛

نتيجة الدالة من نوع عدد صحيح وهو عدد فقرات أو سجلات الملف المخزن في اسطوانة التخزين، فإذا كانت النتيجة صفر فتعني أن الملف فارغ.

١٣ــ٣ استخدام الملفات

USING FILES

قبل استخدام الملفات يجب أن يخصص الملف باستخدام الاجراء «خصص»، ويجب فتح الملف قبل أي عملية ادخال أو اخراج بيانات الملف باستخدام احد الاجرائين «افتح_م_ق» أو «افتح_م_ك» حيث يوضع مؤشر الملف على الفقرة الأولى منه. يمكن توسيع الملف من نهايته أو اضافة فقرات اضافية بوضع المؤشر في آخر الملف ثم كتابة الفقرات المرغوب في اضافتها كما ذكرنا سابقا. ويجب اغلاق الملفات مباشرة بعد الإنتهاء من جميع العمليات عليها باستخدام الاجراء «اغلق» والا فسوف يسبب ضياعا للبيانات وفقدانها من اسطوانة التخزين.

```
برنامج ادخال_بيانات_بضاعة؛
صنف
        اسم = سلسلة [٣٥]؛
        استيراد = سجل
                اسم_البضاعة : اسم؛
                كمية، رقم : صحيح؛
            نهاية؛
    متغير
        ملف_استيراد : ملف من استيراد؛
        بيانات      :استيراد؛
        ل، ع    : صحيح؛
بداية
خصص (ملف_استيراد، 'استيراد.عرب ')؛
افتح_م_ك (ملف استيراد)؛
```

```
اكتب_سطر ('كم عدد البيانات المراد ادخالها؟').
اقرأ (ع)؛
مع بيانات اعمل
بداية
    اكتب_سطر ('ادخل اسم البضاعة وعددها :')
    حيث ل:= ١ إلى ع اعمل
    بداية
        اقرأ (اسم_البضاعة، كمية)؛
        رقم: = ل؛
        اكتب (ملف_استيراد، بيانات)؛
    نهاية؛
  نهاية؛
اغلق (ملف_استيراد)؛
نهاية.
```

البرنامج في المثال السابق يؤسس ملف (استيراد.عرب) وهو قائمة بيانات الاستيراد، ويتكون من عدد محدود من السجلات من الصنف (استيراد) ويحدد عددها بالعدد (ع)، وفي هذه السجلات قيم مبدئية يستطيع المستخدم ادخال واخراج البيانات فيما بعد بطريقة عشوائية، كما نستطيع تطوير هذه البرنامج لجعله مثلا يضيف سجلات أو يحذف سجلات وكذلك قراءتها وتعديلها والكثير من العمليات الأخرى التي يمكن اضافتها عند الحاجة لها.

```
برنامج ادخال_بيانات_تجارية؛
صنف
    اسم = سلسلة [١٥]؛
    استيراد = سجل
        نوع_البضاعة : اسم؛
        كمية، رقم : صحيح؛
```

```
نهاية؛
متغير
    ملف_استيراد : ملف من استيراد؛
    بيانات : استيراد؛
    ل، ع، اعلى : صحيح؛
    جواب، ح : رمز؛
بداية
  كرر
    امسح_شاشة؛
    اكتبسطر؛
    اكتب(' [د] - افتح ملف جديد، [ا] - اضافة سجل'،
        ' [ر] - قراءة السجلات، [و] - خروج :')،
    اقرأ_سطر (جواب)؛
    حالة جواب من
(* يفتح ملف جديد ويضيف السجلات في هذا الملف *)
      'د' : بداية
           اعلى := ٠؛
           خصص (ملف_استيراد، 'استيراد.عرب')؛
           افتح_م_ك (ملف_استيرد)؛
        نهاية
(*          يضيف سجلات جديدة إلى سجلات الملف القديم *)
      'أ' : بداية
           خصص (ملف_استيراد، 'استيراد.عرب)؛
           افتح_م_ق (ملف_استيراد)؛
           اعلى := حجم_ملف (ملف_استيراد)؛ { رقم آخر سجل في الملف}
           بحث_عن (ملف_استيراد، اعلى)؛
نهاية؛
```

(* يقرأ جميع السجلات الموجودة في الملف *)

'ر' : بداية

خصص (ملف_استيراد، 'استيراد.عرب')

افتح_م_ق (ملف_استيراد)؛

اقرأ (ملف_استيراد، بيانات)؛

{طالما لم تظهر اشارة نهاية الملف}

طالما عكس ن_م (ملف_استيراد) **اعمل**

مع بيانات **اعمل**

بداية

اكتبسطر؛

اكتب (رقم، '-'، نوع_البضاعة، ' '، كمية)؛

اقرأ (ملف_استيراد، بيانات)؛

نهاية؛

اكتبسطر؛

اكتب (بيانات.رقم، ' - '، بيانات. نوع_البضاعة،

' '، بيانات.كمية)؛

اغلق (ملف_استيراد)؛

نهاية؛

'و' : اكتبسطر (' ... انتهى ...')

والا اكتبسطر (' ... خطأ في الادخال ...')؛

نهاية؛ {حالة}

{اذا كانت كتابة بيانات وليست قراءة}

اذا (جواب <> 'ر') و ((جواب = 'أ') **أو** (جواب = 'د')) **اذن**

بداية

اكتب ('كم عدد البيانات المراد ادخالها ؟')؛

اقرأ_سطر (ع)؛

مع بيانات **اعمل**

```
بداية
    نوع_البضاعة := ''؛ كمية : = ٠؛
    اكتبسطر(' ادخل نوع البضاعة وكميتها :')؛
    حيث ل := ١ إلى ع اعمل
        بداية
            اكتب (' البضاعة رقم ('، ل، ')':')؛
            اقرأ_سطر (نوع_البضاعة، كمية)؛
            اعلى := اعلى +١؛
            رقم := اعلى؛
            اكتب (ملف_استيراد، بيانات)؛
        نهاية؛
    نهاية؛ {م ع}
    اغلق (ملف_استيراد)؛
  نهاية؛
  اذا جواب <> 'و' اذن
    بداية
        اكتبسطر؛ اكتبسطر؛
        اكتب؛('        اضغط أي مفتاح للاستمرار')؛
        اقرأ (لم،ح)؛
    نهاية؛
  حتى جواب = 'و'؛
نهاية.
```

البرنامج السابق يعطي قائمة تحمل ثلاثة اختيارات، فيمكن فتح ملف جديد يسمى (ملف_استيراد. عرب) وادخال سجلات فيه، واذا كان الملف موجودا في الدليل وبه سجلات، فيمكن إضافة سجلات محدودة إلى سجلات الملف

مبتدئا من نهاية آخر سجل، كما يمكن قراءة سجلات هذا الملف،ويوضح البرنامج ايضا كيفية استعمال الاجراء «بحث_عن»، والاجراء «حجم ملف».

١٣ــ٤ ملفات النص

TEXT FILES

ملفات النص مختلفة قليلا عن الملفات التي نعرفها، فهي ليست فقط قيم متتابعة من نوع واحد... ولكنها مصممة من أسطر، وكل سطر يحتوي على رموز، وتوجد في نهاية كل سطر علامة تبين موضع نهاية السطر، وتوجد أيضا علامة أخرى في نهاية النص تبين موضع هذه النهاية في الملف. ان معالجة ملف النص لا تتم بطريقة عشوائية وانما بطريقة متسلسلة ومتابعة (**SEQUINTIAL**) ومعنى ذلك.. أن مؤشر الملف يجب أن يمر بجميع الرموز واحدا بعد الآخر ولا يمكن قراءة رمز قبل الآخر.

١٣ــ٤ــ١ العمليات على ملفات النص

OPERATIONS ON TEXT FILES

قبل اجراء عمليات المعالجة في ملفات النص، يجب تعريف ملف النص كتعريف نوع قياسي بالكلمة المحجوزة «**نص**»، وبعد ذلك يخصص الملف ثم يفتح باستعمال الاجراءات السابقة للملفات.

الاجراء «افتح_م_ك» يعمل على انشاء ملف نصي جديد، ولا يمكن اجراء أي عملية الا عملية اضافة فقرات جديدة في نهايته، اما الاجراء «افتح_م_ق» فيفتح ملف نصي قديم والعملية المسموحة بعد ذلك هي قراءة الملف. بعد الانتهاء من المعالجة يجب اقفال الملف، وتضاف اليه علامة نهاية الملف (**CTRL Z**) آليا.

وعملية ادخال واخراج الرموز من ملف النص تتم باستعمال الاجرائين القياسيين «اقرأ» و «اكتب» أما معالجة الأسطر فتتم باستعمال الاجراءات (اقرأسطر، اكتب_سطر، ن_س).

١٣ـ٤ـ٢ الاجراءات والدوال على ملفات النص

PROCEDURES AND FUNCTIONS TEXT FILES

بالاضافة الى الاجراءات التي شرحناها، توجد أيضا دوال واجراءات خاصة بمعالجة ملفات النص.

اقرأ_سطر

تركيب الجملة : اقرأ_سطر (اسم_ملف)؛

الاجراء ينقل مؤشر الملف من موضعه على السطر الى بداية السطر التالي.

اكتب سطر

تركيب الجملة : اكتب_سطر (اسم_ملف)؛

يضع علامة نهاية السطر ثم ينقل المؤشر إلى بداية السطر التالي.

ن_س

تركيب الجملة : ن_س (اسم_ملف)؛

نتيجة الدالة من نوع منطقي وهي صحيح اذا كان موضع موضع مؤشر الملف موضوعا على علامة نهاية السطر وعلامة نهاية الملف، وخطأ اذا كان المؤشر في أي مكان آخر.

بحث_ن_س

تركيب الجملة : بحث_ن_س (اسم_ملف)؛

الدالة أيضا من نوع منطقي، وهي مشابهة للدالة «ن_س»، الا أنها تحذف رمز القفز <TAB> والفراغات قبل اختبار علامة نهاية السطر.

بحث_ن_م

نتيجة الدالة من نوعَ منطقي، وهي مشابهة للدالة «ن_م»، الا أنها تحذف علامة القفز <TAB> وعلامة نهاية السطر والفراغات قبل اختبار علامة نهاية الملف.

البرنامج الآتي يوضح طريقة استعمال بعض من هذه الاجراءات والدوال فيعمل على قراءة ملف نصي ونقله إلى ملف آخر.

```
برنامج انقل ملف؛
    متغير
        ملف_١، ملف_٢     :نص؛
        سطر              :سلسلة [٢٥٥]؛
        قديم، جديد       :سلسلة [٢٠]؛
بداية
    اكتب (' ادخل اسم الملف القديم:')،
    اقرأ_سطر (قديم)؛ اكتب_سطر؛
    اكتب ('ادخل الملف الجديد:')؛
    اقرأ_سطر (جديد)؛
    خصص (ملف_١، قديم)؛ خصص (ملف_٢، جديد)؛
    افتح_م_ق (ملف_١)؛
```

```
افتح_م_ك (ملف_٢)؛
طالما عكس ن_م (ملف_١) اعمل
بداية
    اقرأ_سطر (ملف ١، سطر)؛
    اكتب_سطر (ملف_٢، سطر)؛
نهاية؛
اغلق (ملف_١)؛ اغلق (ملف_٢)؛
نهاية.
```

١٣ــ٥ الوحدات المنطقية

LOGICAL DEVICES

الوحدات المنطقية عبارة عن وحدات فرعية من الحاسب الآلي مثل لوحة المفاتيح وآلة الطباعة والمعدل، وتعمل هذه الوحدات على ادخال واخراج البيانات من البرنامج إلى نظام التشغيل الأساسي والعكس، وتعامل هذه الوحدات في البرنامج كمعاملة ملفات النص، إلا ان هناك فرقا بسيطا في معالجة بعض اجراءات ودوال النص، فالاجرائين «افتح_م_ق»، و «افتح_م_ك» ليس في وظيفتهما فرق اذا نفذا على وحدة منطقية، والاجراء «اغلق» لا يعمل شيئاً، ومحاولة مسح وحدة منطقية، سوف تسبب أخطاء في الادخال أو الاخراج للنظام. وتعمل الدالتان «ن_م» و «ن_س» بطريقة مختلفة عن عملهما في النص، فهما تعملان على تفقد حدوث علامة نهاية الملف وعلامة نهاية السطر، فتفحصان الرمز القادم قبل عمل أي اجراء، أما في الوحدات الطرفية فالدالتين تفحصان آخر رمز تمت قراءته بدلا من الرمز التالي، ونستنتج من ذلك ان في الوحدات المنطقية لا نستطيع النظر إلى الرموز المقبلة كما في ملفات النص وانما إلى الرموز الحالية وفيما يلي قائمة الوحدات المنطقية المتداولة.

ن_ط : وحدة النهاية الطرفية، فالخارج من البيانات يرسل الى وحدة الاخراج الطرفية في نظام التشغيل الأساسي (الشاشة)، أما الداخل من البيانات فيرسل من وحدة الادخال الطرفية (لوحة المفاتيح).

شاشة : يرسل الخارج من البيانات إلى وحدة الاخراج الطرفية في نظام التشغيل الأساسي (الشاشة)، والداخل من البيانات يرسل من وحدة الادخال الطرفية (لوحة المفاتيح). والفرق بين الوحدة (ن_ط) والوحدة (شاشة) أن الأولى توفر ذاكرة وسيطة للداخل من البيانات أي أنه عند استعمال اجرائي الادخال (أقرأ، أقرأ_سطر) في البرنامج ومخصصة إلى «ن_ط» فان الوحدة سوف تدخل سطرا كاملا إلى الذاكرة الوسيطة ويستطيع المستخدم تعديل وتغيير هذا السطر طالما أنه موجود في الذاكرة الوسيطة، كما تظهر الرموز الداخلة على الشاشة الا رموز التحكم

ل_م : وحدة لوحة المفاتيح لادخال البيانات فقط، فالبيانات ترسل من وحدة الادخال الطرفية في نظام التشغيل عادة لوحة المفاتيح، ولا تظهر الرموز الداخلة على الشاشة.

طباعة : وحدة الطباعة لاخراج البيانات فقط، فالبيانات ترسل إلى وحدة الطباعة في نظام الشتغيل وهي غالبا الآلة الطابعة.

مساعد : الوحدة الخارجية، تعرف هذه الوحدة حسب نظام التشغيل المستعمل وهي عادة في نظام تشغيل الحاسب الشخصي «COMI».

مستعمل : وحدة المستعمل، فالخارج من البيانات يرسل إلى اجراء المستعمل للاخراج، والداخل يرسل إلى اجراء المستعمل للادخال، هذه الاجراءات سوف نشرحها مفصلة في الأبواب المقبلة.

١٣ـ٦ الملفات القياسية

STANDARD FILES

الملفات القياسية عبارة عن ملفات نص معرفة داخل المترجم ومخصصة إلى وحدة منطقية معينة وجاهزة للمعالجة، ولهذا نجد أن المبرمج قد وفر بعضا من جمل واجراءات البرنامج كفتح الملفات واقفالها، فاستخدام الاجراءات (افتح_م_ق)، افتح_م_ك، اغلق) غير مسموح به اطلاقا على الملفات القياسية، وعند قراءة سطر من الذاكرة الوسيطة بدون تحديد متغير الملف المطلوب في معطيات الاجراء فان سطرا سوف يدخل دائما، ويستطيع المستعمل ايقاف القراءة بالضغط على المفتاح <RETURN> حتى لو .بقي بعض الرموز في الذاكرة ويخزن النظام في نهاية السطر الرمز «CTRL Z» ليحدد نهاية السطر المقروء حتى ولو كان فارغا، وللسيطرة على هذه القراءة الاجبارية راجع الجزء المخصص للموجهات، وهذه الملفات القياسية هي:

الوحدتين المنطقيتين «ن_ط» أو «شاشة».

اخراج : وهو الملف الرئيسي لاخراج البيانات، ومخصص أيضا لاحدى الوحدتين المنطقيتين «ن_ط» أو «شاشة».

ن_ط : مخصص لوحدة النهاية الطرفية «ن_ط».

شاشة : مخصص لوحدة الشاشة «شاشة».

ل_م : مخصص لوحدة المفاتيح «ل_م».

طباعة : مخصص لوحدة الطباعة «طباعة».

مساعد : مخصص للوحدة الخارجية «مساعد».

مستعمل : مخصص لوحدة المستعمل «مستعمل».

في المثال التالي سوف يقرأ البرنامج سطرا مخفيا ن الرموز ينتهي بالمفتاح «RETURN» ثم يرسله مباشرة إلى آلة الطباعة.

```
برنامج ارسل_طباعة_سطر؛
متغير
    سطر :سلسلة [٨٠]؛
بداية
    اكتب_سطر ('اكتب السطر المراد طبعه')؛
    اقرأ (ل_م، سطر)؛
    اكتب_سطر (طباعة، سطر)؛
نهاية.
```

يجب ان نلاحظ أن آلات الطباعة تختلف انواعها بين مستخدم وآخر ولذلك تعتبر من مسئولية المستخدم عملية تجهيز آلته للطباعة.

١٣ـ٧ ادخال واخراج النصوص

TEXTS INPUT AND OUTPUT

لادخال واخراج النصوص في هيئة مقروءة نستعمل ملفات النص كما شرحنا سابقا، فقد يكون ملف النص مخصص لاحدى الوحدات المنطقية التي ذكرناها ولهذا يوجد اجراءات قياسية مصممة في المترجم لعمل هذه العمليات، وهذه الاجراءات نوضحها كالآتي:

اقرأ

تركيب الجملة :

١) اقرأ (اسم ملف، م١، م٢، م٣، ...)؛

٢) اقرأ (م١، م٢، م٣، ...)؛

يسمح هذا الاجراء بادخال رموز وسلاسل رمزية وأرقام، حيث (م١، م٢، م٣، ...) عبارة عن متغيرات من صنف (سلسلة، رمز، حقيقي، صحيح) ففي التركيب الأول يتم ادخال البيانات بقراءتها من ملف النص المخصص

(اسم_ملف)، ويقرأ البرنامج المتغيرات واحدا بعد الآخر فيقرأ المتغير الأول ثم يحذف جميع الفراغات إلى أن يقرأ المتغير الذي يليه فإذا كان عدد المتغيرات في معطيات الاجراء أكثر مما يوجد في السطر فإن خطأ سوف يحدث في الدخول والخروج، أما في التركيب الثاني فيقرأ البرنامج من لوحة المفاتيح وهي الاختيار النظامي فتدخل البيانات حسب ترتيبها حيث تخزن في الذاكرة الوسيطة، وتعتبر الذاكرة الوسيطة مصدر الدخول، والحد الأعلى والنظامي لطول السطر هو ١٢٧ رمزا كما ان هناك بعض مفاتيح ادخال مسموح بها غير مفاتيح ادخال الرموز وهي:

(BACKSPACE) ‘ (DEL)

يؤدي المفتاحان وظيفة واحدة وهي الرجوع خانة واحدة إلى الوراء وحذف الرمز الموجود في هذه الخانة اذا كان موجودا.

(CTRL . X) (ESC)

يعملان على الرجوع إلى بداية السطر الحالي ومسح جميع الرموز على هذا السطر.

(CTRL D)

يستدعي آخر رمز أدخل في آخر سطر تم ادخاله.

(CTRL M) ‘(RETURN)

ينهيان ادخال البيانات في سطر الادخال ويتم وضع علامة نهاية السطر بعد آخر رمز في الذاكرة الوسيطة، وفي بعض لوحات المفاتيح يوجد المفتاح **(ENTER)** بدلا من المفتاح **(RETURN)**.

(CTRL R)

يستدعي آخر سطر أدخل في الذاكرة الوسيطة.

<CTRL> <ئ>

ينهي ادخال البيانات ويضع علامة نهاية ملف بعد آخر رمز.

اقرأ_سطر

تركيب الجملة : ١) اقرأ_سطر (اسم_ملف، م١، م٢، م٣م ٠٠٠)؛

٢) اقرأ_سطر (م١، م٢، م٣، ٠٠٠)؛

هذاالاجراء مشابه للاجراء «اقرأ»، الا ان الفرق هو بعد قراءة آخر معطى فان المؤشر سوف يهمل بقية السطر ويقفز الى بداية السطر الذي يليه، كما نستطيع الغاء المتغيرات لجعل المؤشر يقفز الى بداية السطر التالي بدون ان يقرأ شيئا.

اكتب

تركيب الجملة : ١) اكتب (اسم_ملف، م١، م٢، م٣، ٠٠٠)؛

٢) اكتب : (م١، م٢، م٣، ٠٠٠)؛

يستخدم هذا الاجراء في عمليات الاخراج، والمعطيات (م١، م٢، م٣، ٠٠٠) تتكون من متغيرات من صنف (رموز، سلاسل رموز، اعداد حقيقية وصحيحة، منطقي). التركيب الأول يرسل المعطيات إلى الملف المخصص (اسم_ملف) أما في التركيب الثاني فيرسلها إلى الوحدة النظامية وهي الشاشة.

تشكل صياغة المتغيرات بالاعتماد على نوع المتغير، وتمثل الرموز الآتية:

ل،ص،ع : اعداد صحيحة.

ي : عدد حقيقي.

ر : رمز.

س : سلسلة رموز.

م : منطقي.

وفيما يلي طريقة صياغة وإخراج كل نوع من المعطيات باستخدام الرموز السابقة كالآتي:

المعطيات	اخـــــراج
ر	الرمز (ر)
ر:ع	الرمز (ر) بعدد خانات (ع).
س:	السلسة (س).
س:ع	السلسلة (س) بعدد خانات (ع).
م	يعتمد على القيمة المنطقية (صح، خطأ).
م:ع	يعتمد على القيمة المنطقية (صح، خطأ)، بعدد خانات (ع).
ل	القيمة العددية للمتغير (ل).
ل:ع	القيمة العددية للمتغير (ل) وعدد خانات (ع٠).
ي	القيمة العددية للمتغير (ي)، ويخصص ١٨ خانة مستخدما العلامة العشرية، ويكون الشكل اذا كان: (ي>٠,٠)،*ق # # # # # ## # # # # ،+ && (ي<٠,٠) *ق # # # # #. # # # # # ،- & حيث الرمز# يمثل رقما والرمز(&) يمثل فراغا، والحرف (ق) يمثل القوة، والرمز(*) يمثل اشارة العدد.
ي : ع	القيمة العددية للمتغير (ي)، وعدد خانات (ع).
ع:ع:ص	القيمة العددية للمتغير (ي)، وعدد خانات (ع) للعدد الصحيح ، وعدد خانات (ص) لتمثل الخانات العشرية.

ويجب ملاحظة ان الاشكال السابقة تكون معدلة من اليمين بعدد الخانات المخصصة لها، وتملأ الخانات بفراغات اذا كانت القيم أقل من عدد الخانات.

اكتب_سطر

تركيب الجملة ١) اكتب سطر (اسم_ملف، م١، م٢، م٣، ...)؛

٢) اكتب_سطر (م١، م٢، م٣، ...)؛

هذا الاجراء مشابه للاجراء «اكتب»، الا أنه يقفز إلى بداية السطر الذي يليه بعد كتابة متغيراته، ويمكن الغاء المتغيرات لكتابة سطر فارغ.

المثال التالي يوضح طريقة استعمال بعض اجراءات الدخول والخروج، فهو يقرأ ملف نص ويكتبه على الشاشة.

```
برنامج ادخل_اخرج؛
    متغير
        اسم : سلسلة [٢٠]؛
        اسم_ملف : نص؛
        سطر : سلسلة [٨٠]؛
بداية
    اكتب ('أكتب اسم الملف:')؛
    اقرأ_سطر (اسم)؛
    خصص (اسم_ملف، اسم)؛
    افتح_م_ق (اسم_ملف)؛
    طالما عكس ن_م (اسم_ملف) اعمل
    بداية
        اقرأ_سطر (اسم_ملف، سطر)؛
        اكتب_سطر (سطر)؛
    نهاية؛
    اغلق (اسم_ملف)؛
نهاية.
```

الملفات الغير مصنفة

UNTYPED FILES

الملفات الغير مصنفة عبارة عن قنوات منخفضة في النظام، وتستخدم للوصول المباشر والسريع إلى أي ملف في اسطوانة التخزين، وتستخدم هذه الملفات سجلا سعته ١٢٨ بايت. تتم عمليات الادخال والاخراج فيها بنقل البيانات والقيم مباشرة من الاسطوانة إلى المتغير موفرا المكان المخصص على ذاكرة القطاع أو الذاكرة الوسيطة والمستخدمة في الملفات المصنفة ولذلك فهي تحجز أقل مكانا في الذاكرة مما تحجزه الملفات المصنفة، كما انها متوافقة مع أي ملف، ويفضل استخدام هذا النوع عندما لا يتضمن البرنامج عمليات ادخال واخراج بيانات مثل عمليات الغاء وتسميه الملفات، وهذا الملفات الغير مصنفة غير منظمة على هيئة سجلات حيث كل فقرة بيانات تتكون من سجل، فهي لا تعمل أي اعتبار على شكل وتنظيم البيانات.

وتعرف الملفات الغير مصنفة في البرنامج باستخدام الكلمة المحجوزة «**ملف**» فقط ولا يتبعها أي صنف كالآتي:

متغير

اسم_ملف : **ملف**؛

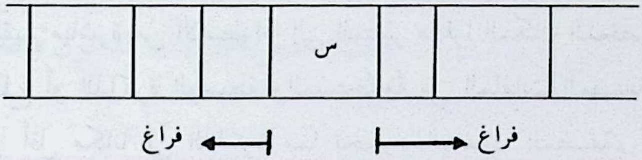
١٣ـ٩ اقرأ_كتلة / اكتب_كتلة

READBLOCK / WRITEBLOCK

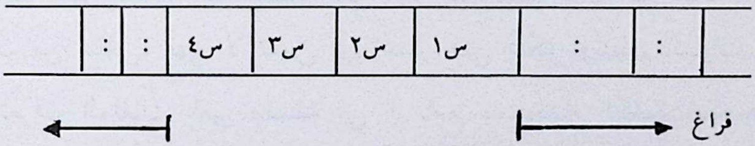
ان جميع الاجراءات والدوال القياسية على الملفات يمكن تطبيقها على الملفات الغير مصنفة ماعدا الاجرائين (اقرأ، اكتب)، فقد تم استبدالها بالاجرائين (اقرأ_كتلة، اكتب_كتلة)، والمقصود بالكتلة هنا. ! أي كتلة ذاكرة،

وهي مقسمة إلى مجموعة سجلات وكل سجل يحتوي على مجموعة من

الارقام والرموز والسلاسل الرمزية، وتتم معاملة هذه السجلات ككتلة واحدة وقد تتساوى هذه الكتل في عدد السجلات وطولها أو تختلف.



كتلة تتكون من سجل واحد (س)



كتلة تتكون من أربعة سجلات

تركيب الجملة:

١) اقرأ_كتلة (اسم_ملف، م، ع)؛

اكتب_كتلة (اسم_ملف م، ع)؛

٢) اقرأ_كتلة (اسم_ملف م، ع، ن)؛

اكتب_كتلة (اسم_ملف م، ع، ن)؛

يعمل الاجراء على قراءة ملف من اسطوانة التخزين مباشرة حيث يكون مؤشر الملف موضوعا على وحدة التخزين «بايت» الأولى التي يحتجزها المتغير (م) في السجل، والمتغير (ع) يعرف العدد ١٢٨ وهو عدد خلايا التخزين

«بايت» في السجل الواحد والتي سوف تنقل من الاسطوانة إلى المتغير (م)، فيجب التأكد ان (م) قادرة على اتساع البيانات المنقولة اليها، اما المتغير (ن) فهو اختياري ويمثل عدد الوحدات التي قرأت وتم انتقالها. يجب اعداد الملف قبل اجراء أي عملية عليه بتخصيصه ثم فتحه وبعد الانتهاء منه يقفل لضمان نهاية المعالجة.

المثال الآتي يوضح الطريقة الصحيحة لاستعمال الاجرائين السابقين في الملفات الغير مصنفة، وهو يعمل على نقل ملف قديم من أي نوع الى ملف آخر جديد.

```
برنامج انقل_ملف_قديم؛
  ثابت
      حجم_السجل = ١٢٨؛
      حجم_الذاكرة = ٢٠٠؛
  متغير
      قديم، جديد : ملف؛
      اسم١، اسم٢ : سلسلة [٢٠]
      الذاكرة : منظومة [١.. حجم_السجل، ١.. حجم_الذاكرة] من بايت؛
      عدد_سجل : صحيح؛
بداية
  اكتب (' اسم الملف القديم:')؛
  اقرأ_سطر (اسم١)؛
  اكتب (' اسم الملف الجديد:')؛
  اقرأ_سطر (اسم٢)؛
  خصص (قديم، اسم١)؛
  خصص (جديد، اسم٢)؛
  افتح_م_ق (قديم)؛
```

```
افتح_م_ق (جديد)؛
كرر
  اقرأ_كتلة (قديم، الذاكرة، حجم_الذاكرة، عدد_سجل)؛
  اكتب_كتلة (جديد، الذاكرة، عدد_سجل)؛
حتى عدد_سجل = ٠؛
  اغلق (قديم)؛
  اغلق (جديد)؛
نهاية.
```

١٣ـ١٠ فحص د/خ

I/O CHECKING

تستخدم موجهة المترجم {م} في التحكم بإصدار الاخطاء الناتجة عن الادخال والاخراج. فاستعمال الموجهة الموجبة {$م+} تجعل المترجم ينادي روتين فحص الادخال والاخراج بعد كل عملية د/خ، وعند حدوث خطأ فإن المترجم سوف يتوقف ويعطي التحذير الخاص بنوع الخطأ.

أما عند استعمال الموجهة السالبة {$م- } فإن الفحص لن يتم، وسيستمر المترجم في الترجمة حتى اذا كان هناك خطأ في د/خ وسوف يهمل أي عملية د/خ، ويضعها في حالة توقف تام حتى تأتي عملية استعمال الدالة «**نتيجة_د_خ**» في البرنامج، فيقوم المترجم باعادة حالة الادخال والاخراج إلى طبيعتها ومن ثم يمكن اجراء عمليات د/خ، مرة أخرى، الدالة (نتيجة_د_خ) تكون نتيجتها صفرا اذا كانت عملية د/خ ناجحة، وأي رقم آخر اذا كان هناك خطأ في آخر عملية د/خ، ويجب وضع هذه الدالة في البرنامج عندما يتوقف بدون نتيجة مقنعة تمت في عملية د/خ وخاصة بعد استعمال الاجراءات القياسية التالية:

اغلق	**سلسل**	**الحق**
امسح	**اقرأ**	**اسم_دليل**
ابحث	**اقرأ_سطر**	**اعمل_دليل**
ارسل_الى_ملف	**غير_اسم**	**احذف_دليل**
اقرأ_كتلة	**افتح_م_ق**	**غير_دليل**
اكتب_كتلة	**افتح_م_ك**	**نفذ**
اكتب	**اكتب_سطر**	

المثال الآتي يوضح عمل الدالة نتيجة_د_خ، فيقوم بالسؤال المستمر عن ملف موجود حتى تنجح عملية فتح الملف.

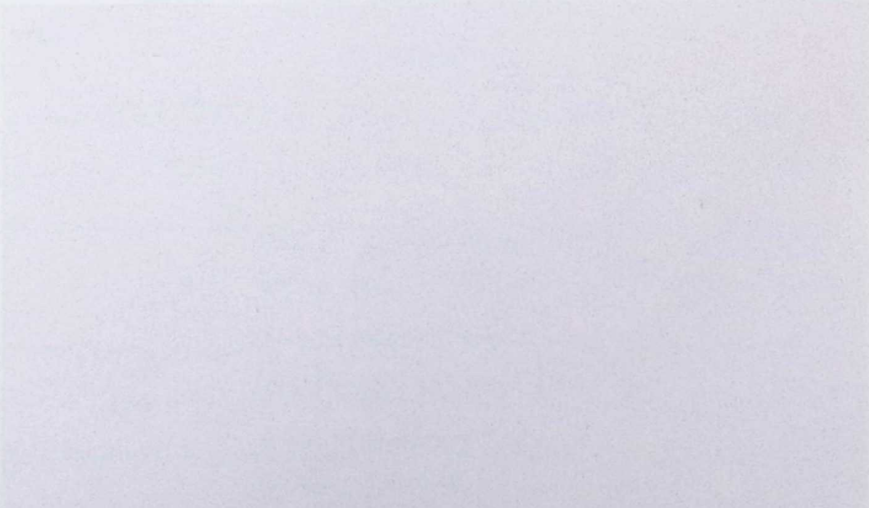
مثال:

```
اجراء فتح_ملف،
  بداية
    كرر
      اكتب (' اسم الملف الموجود' :)؛
      اقرأ سطر (اسم)؛
      خصص (اسم_ملف، اسم)؛
      {$م+}افتح م ق (اسم ملف) {$ م -}؛
      موجود := (نتيجة_د_ح =٠)؛
      اذا عكس موجود اذن
        اكتب_سطر (' الملف غير موجود :'،اسم_ملف)؛
    حتى موجود؛
  نهاية؛
```

OOOOOOO

[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]

[illegible]



[illegible]

الباب الرابع عشر

المؤشـــرات

تمهيــد

ان أنواع المتغيرات التي تحدثنا عنها في الأبواب السابقة عبارة عن متغيرات **ساكنة**، فأحجامها وأشكالها محددة، وتتواجد في جميع الكتلة التي تعرف فيها خلال عملية التنفيذ، ولهذا تأتي الحاجة الى استعمال البيانات الهيكلية (STRUCTURED DATA) والتي تختلف أحجامها وأشكالها خلال عملية التنفيذ. المتغيرات **المتحركة** تستطيع أن تؤدي هذا الغرض فهي تولد وتتكون في البرنامج عند الحاجة اليها ثم تلغى بعد الانتهاء منها.. أي ان البرنامج قد يستعمل مكان واحد في الذاكرة لمتغير متحرك ، وبعد الانتهاء منه والغائه قد يستعمل نفس المكان لمتغير متحرك آخر. وتتميز المتغيرات المتحركة بأنه لا يمكن تعريفها بطريقة واضحة ولا الإشارة اليها مباشرة باسم التعريف مثل المتغيرات الساكنة، ولكن يستخدم متغير خاص يسمى **المؤشر** يحمل عنوان المتغير المطلوب في الذاكرة ولهذا فهو مؤشر على المتغير المطلوب.

١٤ــ١ تعريف المؤشرات في البرنامج

POINTERS DEFINETION

يعرف نوع المؤشر باستعمال رمز المؤشر ^ يليه صنف المتغير المتحرك من الأصناف القياسية (حقيقي، رمز، سجل، ...)، ويستخدم المؤشر غالبا في السجلات، فهو يسهل تنظيم هيكلها فبعد تعريف حقول السجل يضاف

اليها حقل إضافي خاص بمؤشر ليؤشر على السجل الذي يليه، وتسمى هذه البنية من السجلات «**عقدة**» (NODE)، وعند ربط عدة عقد ببعضها البعض بواسطة مؤشراتها فاننا نحصل على بنية هيكلية جديدة للبيانات تشبه القائمة ولكنها قائمة متحركة أي ان الحد الأقصى لسعتها غير معروف، فهذه القائمة قد تنمو بحجم الذاكرة المتوفرة في الحاسب الآلي (RAM) وتسمى **بالقوائم المتصلة** (LINKED LISTS).

مثال :

صنف

مؤشر_شخصي = ^سجل_شخصي؛

سجل_شخصي = **سجل**

اسم : **سلسلة** [٣٠]؛

العمر : صحيح؛

الحالة_الاجتماعية : (متزوج، اعزب، ارمل)؛

مؤشر_المجاور : مؤشر_شخصي؛

نهاية؛

متغير

اول_شخص، آخر_شخص، شخص_جديد: مؤشر_شخصي؛

في المثال السابق يتضح كيفية تعريف السجلات المرتبطة بالمؤشر وتكوين عقد متكاملة لتحتوي على هذه السجلات، فالمؤشر (مؤشر_شخصي) يؤشر على السجل (سجل_شخصي)، فهو يعرف عقدة واحدة تحتوي على سجل واحد، ويحتوي كل سجل على الحقل الاضافي (مؤشر_المجاور) وهو من نوع (مؤشر_شخصي) ليستخدم في الاشارة إلى العقدة التي تليه، والمتغيرات (اول_شخص، آخر_شخص، شخص_جديد) عبارة عن متغيرات مؤشرات تؤشر على عقد تحتوي على سجلات من نوع (سجل_شخصي).

١٤ـ٢ الكومة

THE HEAP

قبل استخدام المؤشر يجب أن يوجد متغيرات يخصص لها مواقع في الذاكرة باستعمال الاجراء «**جديد**» حيث يصاغ باعطائه معطيا واحدا من نوع مؤشر إلى متغير من أي نوع من الأنواع التي نريد تداولها.

مثال :

جديد (أول_شخص)؛

يجعل هذا الاجراء المؤشر (أول_شخص) يؤشر بطرقة حركية على عنوان سجل من نوع (سجل_شخصي) في الذاكرة ويخصص موضعا في الذاكرة ليتسع لهذا السجل.

المتغيرات المتحركة التي يصنعها الاجراء «جديد» تخزن في بنية تخزين هيكلية تشبه **الرزمة** (STACK) في بنيتها الهيكلية.. فهي تشبه كتب موضوعة فوق بعضها في درج المكتب فالكتاب الموضوع أخيرا يمكن أخذه أولا والكتاب الموضوع أولا ــ لا يمكن أخذه الا أخيرا، وتسمى هذه البنية الهيكلية **بالكومة** (HEAP)، وتحدد سعتها بسعة ذاكرة الحاسب الآلي، ويتحكم النظام في هذه الكومة بمؤشر يسمى **مؤشر_ذ_س_م** (HEAPTR) في البرنامج ولكن سنسميه مؤشر الكومة في الشرح، فعند بداية البرنامج يحتوي هذا المؤشر مبدئيا على أول وحدة تخزين شاغرة (بايت) في الذاكرة، ومتغيراته لا تحمل بيانات كالتي تحملها المتغيرات الأخرى، وانما تحمل العنوان الذي يؤشر على المتغير المخزن في الكومة. وتخصص المؤشرات الى بعضها بشرط أن تكون من صنف واحد، وتقارن اذا كانت من صنف واحد باستعمال العلاقتين (=،<>) ونتيجتها منطقية (صح، خطأ)، وتستخدم قيمة المؤشر **فارغ**

(NIL) لتؤشر على القيمة الفارغة وهي لا شيء، وتستعمل دائما عندما لا يوجد متغير محدد موقعه. كما تقارن هذه القيمة مع قيم المؤشرات الأخرى فهي متوافقة مع جميع أنواع المؤشرات.

١٤ـ٣ اجراءات ودوال المؤشرات

PROCEDURES AND FUNCTIONS ON POINTERS

بالاضافة للاجراء «جديد»، للمؤشرات ودوال قياسية صممت في النظام لتساعد المبرمج على اختصار برنامجه وتوفير وقته وهي:

(علامة)، (اطلق)

تركيب الجملة : علامة (م)؛

اطلق (م)؛

عندما ينتهي البرنامج من استخدام المتغيرات الحركية فالاجرائين (علامة، اطلق) يعملان على استرداد الذاكرة التي حددت لهذه المتغيرات، فيقوم الإجراء (علامة) بتخصيص قيمة مؤشر الكومة الى متغير المؤشر (م)، أما الاجراء (اطلق) فيحدد مؤشر الكومة إلى عنوان متغير المؤشر (م) الذي علّم من قبل بالاجراء (علامة) فيحرر جميع الأماكن التي حددت من موقع هذا المؤشر وما فوقه أي أن الاجراءان يقومان بتنظيف الكومة من المتغيرات التي تأخذ مكانا فيها ولا حاجة لها.

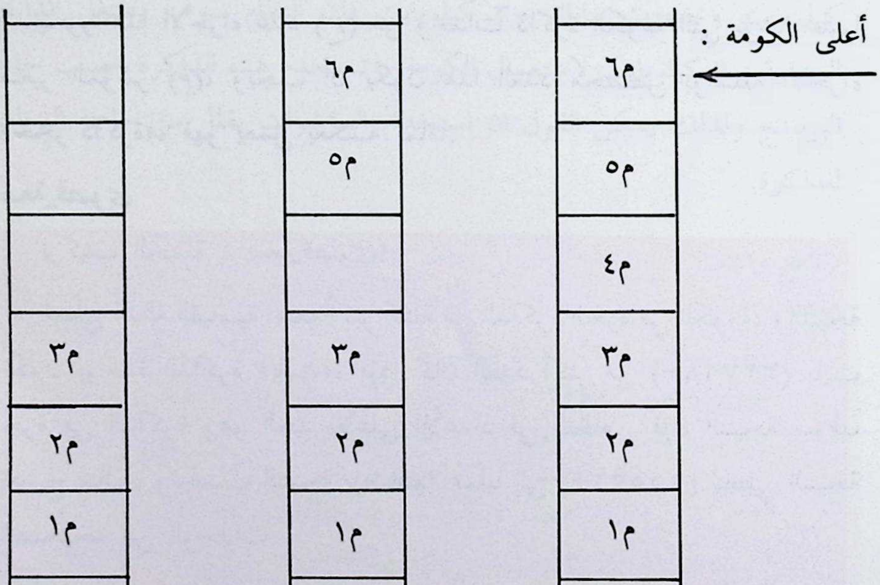
احذف_ذاكرة

تركيب الجملة : احذف_ذاكرة (م)؛

بدلا من استعمال الاجرائين (علامة، اطلق)، فالاجراء القياسي «احذف_ذاكرة» يؤدي نفس الغرض في تحرير الأمكنة الشاغرة في الكومة ويحررها

بطريقة تختلف عن طريقة الاجرائين (علامة، اطلق)، ولهذا لا يمكن خلطها معا في البرنامج والا سوف تؤدي إلى نتائج غير متوقعة، فهو يحرر المكان الذي يشغله متغير المؤشر (م) فقط، بينما الاجراءان (علامة، اطلق) يحرران الأمكنة من متغير المؤشر (م) فما فوقه.

لنفرض أن لدينا بعض المتغيرات المخزنة في الكومة، فالرسم التالي يوضح طريقة عمل الاجراءات الثلاثة (علامة، اطلق، احذف_ذاكرة) في كومة تحتوي على مجموعة من المتغيرات:



الكومة   احذف_ذاكرة (م٤)   علامة (م٤)
اطلق (م٤)

حجز_ذاكرة

تركيب الجملة : حجز_ذاكرة (م، ع)؛

يخصص هذا الاجراء عدد (ع) من وحدات ذاكرة الكومة لمتغير المؤشر (م)، ويختلف عن الاجراء (جديد) بأنه يتحكم بعدد وحدات الذاكرة المطلوبة، بينما الاجراء (جديد) يخصص عدد وحدات بحجم المكان المطلوب لمؤشره.

ذاكرة_حرة

تركيب الجملة : ذاكرة_حرة (م، ع)؛

يحرر هذا الاجراء عدد (ع) من وحدات ذاكرة الكومة التي يؤشر عليها متغير المؤشر (م)، ويجب أن يكون هذا العدد مخصص بواسطة الاجراء «حجز ذاكرة»، فهو يعمل بعكسه تماما.

سعة_قصوى

تركيب الجملة : سعة_قصوى؛

تعطي الدالة القياسية سعة أكبر كتلة من الذاكرة الحرة في الكومة، والنتيجة تكون بوحدة الذاكرة «بايت»، فإذا كان العدد أكثر من (+٣٢٧٦٨) بايت حرة في الذاكرة وهو الحد الأعلى للأعداد في النظام.. فإن النتيجة سوف تصبح سالبة، وتحسب النتيجة بإضافتها عمليا إلى (+٦٥٥٣٦) لتعطي النتيجة الصحيحة من الوحدات.

٤ــ١٤ مميزات المؤشرات

ADVANTAGES OF POINTERS

ان هناك ميزتان تعطي للمؤشر مكانة في البنية الهيكلية، فالميزة الأولى : أن القوائم المتصلة تنمو بطريقة حركية عند تنفيذ البرنامج، وهذه القوائم هي

النوع الوحيد في الذاكرة التي تغير حجمها خلال تنفيذ البرنامج ففي المنظومات مثلا قد لا تمتليء المنظومة كلها بالبيانات ولكنها تحجز الحد الأعلى المخصص لها في الذاكرة.

ثانياً : أن طول البرنامج محدد له ٦٤ كيلوبايت في الذاكرة فقط كحد أعلى ومحدد أيضا ٦٤ كيلوبايت أخرى للبناء الهيكلي (جميع المتغيرات والمنظومات) في البرنامج الواحد، ولهذا فالنظام يقوم باستغلال الذاكرة الفائضة من حجم البرنامج من خلال استعمال الكومة، فمثلا اذا حجز البرنامج ١٨ كيلوبايت وحجزت بياناته ٣٠ كيلوبايت فإن النظام سوف يستغل الذاكرة المتبقية ٨٠ كيلوبايت لاستعمال الكومة، وتستخدم هذه الذاكرة حسب حاجة البرنامج ولذلك يمكن القول ان استخدام المؤشرات يقلل من حاجة البرنامج للذاكرة.

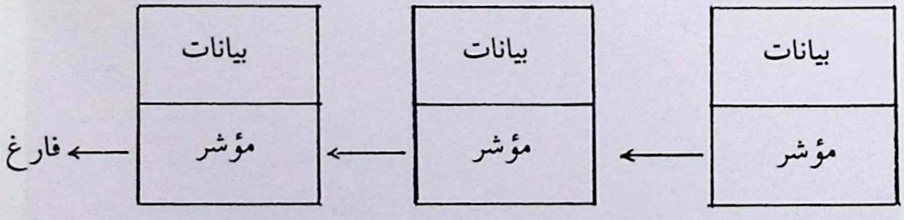
```
برنامج وظائف؛
صنف
    مؤشر_شخصي =  ^  سجل_شخصي؛
    سجل_شخصي = سجل
        اسم          : سلسلة [٣٠]؛
        العمر        : صحيح؛
        الحالة_الاجتماعية    :سلسلة [١٠]؛
        مؤشر         : مؤشر_شخصي؛
    نهاية؛
متغير
    اعلى_الكومة : صحيح؛
    أول_شخص، آخر_شخص، شخص_جديد : مؤشر_شخصي؛
    اسم :سلسلة [٣٠]؛
```

```
بداية
  اول_شخص : =فارغ؛
  علامة (اعلى_الكومة)؛
  كرر
    اكتب ('ادخل الاسم :')؛ أقرأ_سطر (اسم)؛
    اذا (اسم <> '') اذن
    بداية
      جديد (شخص_جديد)؛
      شخص_جديد اسم := اسم؛ اكتب (' العمر : ')؛
      اقرأ_سطر (شخص_جديد ^ .العمر)؛ اكتب ('الحالة_الاجتماعية:')؛
      اقرأ_سطر (شخص_جديد ^ .الحالة_الاجتماعية)؛
      اذا (اول_شخص = فارغ) اذن
        اول_شخص := شخص_جديد
      والا
        آخر_شخص ^ .مؤشر := شخص_جديد؛
        آخر_شخص := جديد؛
        آخر_شخص ^ .مؤشر := فارغ؛
    نهاية؛
  حتى اسم ='')؛ اكتب_سطر؛
  طالما اول_شخص <> فارغ اعمل
  مع اول_شخص^ اعمل
  بداية
    اكتب_سطر (اسم، عمره'،العمر، وهو، الحالة_الاجتماعية)؛
    اول_شخص := مؤشر؛
  نهاية؛
  اطلق (اعلى_الكومة)؛
نهاية.
```

١٤ـ٥ القوائم المتصلة

LINKED LISTS

يعتبر استعمال المنظومات من أسهل الطرق لصنع القوائم ولكن المشكلة الوحيدة أنه يجب تعريف الحد الأعلى من العناصر لنعرف طول القائمة، وبذلك نفقد الكثير من حيز الذاكرة. تعتبر القوائم المتصلة الحل الوحيد والناجح لحل هذه المشكلة، ويمكن بناء العديد من الهياكل وتشكيلها حسب حاجة المبرمج ونوع الأغراض المراد عملها بهذه القوائم، **فالقوائم المتصلة المفردة** تستخدم مؤشرا واحدا في العقدة ليؤشر على السجل الذي يليه وتعمل في مسار واحد. **والقوائم المتصلة المزدوجة**، تستخدم مؤشرين في كل عقدة، فمؤشر يؤشر على العقدة السابقة، والمؤشر على العقدة اللاحقة ولهذا تعتبر قوائم ذات مسارين، والعديد من هذه البنيات الهيكلية التي لا نستطيع ذكرها في نطاق هذا الكتاب، ولكن يستطيع المبرمج مراجعة الكتب المتخصصة في البناء الهيكلي للبرامج (**DATA STRUCTURE**) وتطبيقها على هذه القوائم المتصلة.



(عقدة واحدة)

الشكل السابق يوضح أحد الطرق لتكوين البنية الهيكلية للقوائم المتصلة المفردة، ونلاحظ أن المؤشر في العقدة الأخيرة يؤشر على فارغ، ولذلك يمكن اضافة عقد أخرى بدون الاهتمام في عددها، ويجب المحافظة على تنظيم هذه العقد، فمن مسئولية المبرمج تعريف العقدة التي تبدأ القائمة وتنتهي منها،

ولذلك فالمبرمج قد يحتاج إلى تعريف ثلاثة مؤشرات في البرنامج المحتوي على قوائم متصلة مفردة، فالمؤشر الأول يؤشر على بداية القائمة والمؤشر الثاني يؤشر على العقدة الحالية والمؤشر الثالث يؤشر على العقدة التالية للعقدة الحالية كالآتي:

متغير

أول_سجل،

سجل_حالي،

سجل_تالي : مؤشر_شخصي؛

فعند بداية القائمة لا يوجد مؤشر على (سجل_تالي)، ولذلك يجب اعطاء القيمة المبدئية «فارغ» للمؤشر (أول_سجل)، وعند الرغبة في عمل عقد اضافية فإن البرنامج يجب ان يجري عملية مقارنة على مؤشر (أول_سجل) ليعرف اذا كان اول اتصال للقائمة ام لا.

مثال:

```
اذا أول_سجل = فارغ اذن
    بداية
        جديد (سجل_حالي)؛
        أول_سجل : = سجل_حالي؛
        سجل_حالي ^ . مؤشر : = فارغ؛
    نهاية
والا
    بداية
        سجل_تالي : = سجل_حالي؛
        جديد (سجل_حالي)؛
```

```
سجل_تالي، ^ . مؤشر : = سجل_حالي،
سجل_حالي ^ . مؤشر : = فارغ؛
نهاية؛
```

في المثال السابق، تعمل عملية المقارنة على اختبار ما اذا كان أول اتصال في القائمة، فإذا كان صحيحا فإن البرنامج يصنع عقدة جديدة تحتوي على السجل الحالي ويجعلها العقدة الأولى في القائمة، ثم يخصص القيمة الفارغة لمؤشرها، فإذا تكررت المقارنة بعد ذلك فإن العقدة الأولى لن تكون فارغة، ولذلك سيقفز البرنامج إلى العبارة «والا» حيث يخصص قيمة السجل السابق إلى السجل الحالي ويصنع عقدة جديدة تحتوي على السجل الحالي ويجعل مؤشر السجل السابق يؤشر على السجل التالي ويخصص القيمة الفارغة لمؤشره، وفي هذه الحالة نرى استعمالات المؤشرات الثلاثة.

```
برنامج هيكل_الرزمة؛
    صنف
        مؤشر_عقدة          =  ^ صنف_العناصر؛
        صنف_العناصر = سجل
            اسم : سلسلة [٢٠]؛
            اسم : مؤشر_عقدة؛
        نهاية؛
    متغير
        شاغر، عقدة، قائمة_عقد : مؤشر_عقدة؛
        ح : رمز؛
    (٥ يعمل على وضع سجل واحد على أعلى الرزمة ٥)
اجراء ادفع_عقدة (متغير عقدة، قائمة : مؤشر_عقدة)؛
بداية
    عقدة ^ . سابق : = قائمة؛
```

```
قائمة := عقدة
نهاية؛

(ه بأخذ السجل الموجود على أعلى الرزمة ه)
اجراء اسحب_عقدة (متغير عقدة، قائمة : مؤشر_عقدة)؛
بداية
    اذا قائمة = فارغ اذن
        جديد (عقدة)
    والا
    بداية
        عقدة    := قائمة؛
        قائمة := سابق ^ . سابق
    نهاية
نهاية ؛ (ه اسحب عقدة ه)

(ه يعطي قائمة السجلات التي الرزمة ه)
اجراء اظهر_قائمة (ع : مؤشر_عقدة)؛
بداية
    طالما ع <> فارغ اعمل
    بداية
        اكتب_سطر (ع ^ .اسم)؛
        ع := ع ^ .سابق؛
    نهاية؛

    اكتب_سطر؛
نهاية؛ (ه اظهر قائمة ه)
```

```
(*	يعطي سعة الذاكرة في الرزمة وكذلك الخانات الشاغرة *)
	اجراء اعرض؛
		بداية
			امسح_شاشة؛
			اكتب_سطر ('ذاكرة =', سعة_ذم )؛
			اكتب_سطر؛
			اكتب_سطر (' قائمة الخانات الشاغرة :')؛
			اظهر_قائمة (شاغر)؛
			اكتب_سطر (' قائمة الاسماء :')؛
			اظهر_قائمة (قائمة_عقد)؛
		نهاية، (* اعرض *)

(*	يضيف سجل واحد إلى قائمة الاسماء *)
	اجراء اضف_عقدة؛
		بداية
			اكتب_سطر؛
			اسحب_عقدة (عقدة، شاغر)؛
			اكتب (' ادخل الاسم : ')؛
			اقرأ_سطر (عقدة ^ . اسم)؛
			ادفع_عقدة (عقدة، قائمة_عقد)؛
		نهاية ؛ (* اضف عقدة *)

(*	يلغي سجل واحد من قائمة الأسماء *)
	اجراء الغاء_عقدة؛
		بداية
			اسحب_عقدة (عقدة، قائمة_عقد)؛
			ادفع_عقدة (عقد، شاغر)؛
		نهاية ؛ (* الغاء عقدة *)؛
```

```
بداية (٥ البرنامج الرئيسي ٥)؛
    شاغر := فارغ؛
    قائمة_عقد := فارغ؛
  كرر
    اعرض؛
    اكتب سطر؛
    اكتب (’ [F1]ـــ اضف،          [F2]ـــ إلغاء، [F3]ـــ نهاية’)؛
    اقرأ (ل_م، ح)؛
    اذا ضغط_مفتاح اذن
        اذا (ح = # ٢٧) و (ضغط_مفتاح) اذن
            اقرأ (ل_م، ح)؛
    حالة ح من
        # ٥٩     : اضف_عقدة؛
        # ٦٠     : الغاء_عقدة؛
        # ٦١     : اكتب_سطر (’نهاية’)
    والا اكتب_سطر (’ooo خطأ المفتاح غير معروف ooo’)؛
    نهاية؛
  حتى ح = #٦١
نهاية.
```

OOOOOOO

الباب الخامس عشر

الاجراءات والدوال PROCEDURES AND FUNCTIONS

تمهيد

يتكون البرنامج من كتلة واحدة أو مجموعة من الكتل، ومن أصول البرمجة السليمة تقسيم كتل البرامج الطويلة إلى كتل صغيرة تساعد على الاختصار من طول البرنامج وإلغاء العمليات المكررة، كما تصبح مراجعته وفهمه فيما بعد سهلة ومبسطة وقد تحتوي الكتلة على عدة كتل أخرى، فبعضها قد تكون إما اجراءات أو دوال، وهي عبارة عن **برامج ثانوية** لها مميزاتها الخاصة. فالاجراء يكون في جزء منفرد في البرنامج فينفذ من أي مكان فيه بكتابة اسم الاجراء، والدالة أيضا تنفذ كذلك لكنها تجري حسابات أو تقييم ثم تعطي نتيجة واحدة بعد تنفيذها.

والبرامج الثانوية تختصر البرنامج وتجعله أكثر وضوحا، فإذا كان المبرمج يستخدم جملا معينة وعمليات حسابية بكثرة فمن الأفضل وضعها في برامج ثانوية وكتابتها مرة واحدة بدلا من التكرار واطالة البرنامج، والنظام يحتوي على الكثير من الاجراءات والدوال القياسية ولكنها لا تشمل كل شيء يحتاجه المبرمج ولكنها وضعت للمساعدة في عمليات البرمجة الضرورية ومن خصائص المبرمج الذكي استعمال البرامج الثانوية لتقسيم البرنامج إلى أجزاء رئيسية كل منها يخدم غرضا معينا ومحاولة ربط هذه الأجزاء للوصول إلى الهدف بأقصر الطرق وتصميم برنامج متكامل.

١٥ـ١ المعطيات

PARAMETERS

يغير المبرمج قيم الاجراء أو الدالة بواسطة المعطيات، فهي تتيح للمبرمج أن ينفذ الاجراء أو الدالة بقيم مختلفة وذلك بتمريرها كمعطيات في الجملة، وهناك قائمتان منها، القائمة الأولى تأتي في صدارة البرنامج الثانوي وتوضع بين قوسين وتسمى **قائمة المعطيات الأساسية** وتمرر المعطيات إلى البرنامج في جملة تحمل قائمة معطيات تسمى **قائمة المعطيات الفعلية** ، ويجب أن يكون ترتيب وتعريف المعطيات في كلا القائمتين متساويا، لكن لا يشترط أن تكون أسماء متغيراتها متساوية. ويسمح المترجم بتمرير المعطيات بطريقة **القيم** (**VALUE**) أو طريقة **الإسناد** (**REFERENCE**) أو كليهما معا، وتستخدم غالبا في تنفيذ الاجراءات، وتمثل مدى التأثير على قيم المعطيات الفعلية في البرنامج الثانوي فطريقة القيم تعمل على تنفيذ البرنامج الثانوي بالقيم التي في المعطيات الفعلية ولكن لا يطرأ عليها تغيير بعد الانتهاء من التنفيذ، فالمعطيات الفعلية تمثل معطيات محلية في البرنامج الثانوي، أما طريقة الاسناد فتمثل المعطيات الأساسية على أنها المعطيات الفعلية وتغير قيمتها بعد التنفيذ.

ويميز البرنامج الثانوي المعطيات الفعلية من المعطيات الأساسية بوضع الكلمة المحجوزة «**متغير**» قبل المعطيات الفعلية.

أمثلة:

١)معطيات أساسية بطريقة القيم:

اجراء اجمع_عددين (عدد١، عدد٢ : صحيح)؛

دالة جذر_تربيعي (عدد : صحيح) : حقيقي؛

٢) معطيات أساسية بطريقة الاسناد:

اجراء ترتيب_تنازلي (**متغير** جدول : م)؛

وتمرر المعطيات للبرنامج الثانوي باستعمال المعطيات الأساسية كالآتي:

اجمع_عددين (ع١، ع٢)؛

جذر : = جذر_تربيعي (ع)؛

فقيم المتغيرات (ع١، ع٢) تمرر إلى الاجراء ولكنها لا تتغير بعد تنفيذه، بحيث (عدد١) يأخذ قيمة (ع١) وقيمة (ع٢) تمرر إلى (عدد٢)، كذلك قيمة (ع) تمرر إلى (عدد)، ويعرف المتغير المخصص للدالة كنفس صنف نتيجة الدالة، فالمتغير (جذر) في المثال السابق من صنف صحيح. وفي المثال الثاني إذا عرِّف المتغيران (م،ج) في البرنامج الرئيسي بأنهما من صنف مدى وهو:

صنف

مدى : **منظومة** [١..٩٠٩] **من** صحيح؛

متغير

م، ج : مدى؛

تمرر المعطيات للبرنامج الثانوي كالآتي:

ترتيب_تنازلي (ج)؛

فعند الانتهاء من تنفيذ الاجراء تتغير قيم المنظومة (ج)، ويجب ملاحظة ان البيانات الهيكلية المركبة كالمنظومات لا تعرف داخل معطيات البرنامج الثانوي لصعوبة تمريرها وحجزها الكثير من وحدات الذاكرة، فالأفضل تعريفها في منطقة التصنيف ثم تعرف في المعطيات بأنها من هذا الصنف فيقوم المترجم باعطاء البرنامج الثانوي عنوان البيانات الهيكلية المركبة، فإذا كانت المعطيات تمرر بطريقة القيم، فإن المترجم سيعطي نسخة منها لاستعمالها البرنامج الثانوي لكي لا تتغير البيانات الأصلية.

١٥ــ٢ الاجراءات

PROCEDURES

الاجراءات نوعان : إما قياسية (داخلية) أو معرفة من قبل (المبرمج)، فالقياسية جزء من نظام اللغة ومصممة في داخله، ويجوز تعريف اجراء باسم اجراء آخر قياسي ولكنه يلغي تنفيذ الاجراء القياسي ويعتبر اسم الاجراء خاصا فقط للاجراء المعرف، والأفضل محاولة تجنب حدوث ذلك لكي لا يقع إلتباس في تصميم البرنامج، وتستخدم الاجرءات عندما يريد المبرمج حساب أكثر من قيمة واحدة أو لتقسيم البرنامج إلى أجزاء ثانوية كل منها ينفذ عمليات مستقلة لتجعل من البرنامج أكثر كفاءة وتسهل صيانته.

ويعرف الاجراء كالبرنامج ولكن تستعمل الكلمة المحجوزة (**اجراء**) كبداية بدلا من برنامج، ثم يأتي اسم الاجراء، يليه قائمة المعطيات اذا وجدت.

أمثلة:

اجراء تكرار؛

اجراء نقطة (س، ص : صحيح)؛

اجراء (**متغير** بيان : جدول؛ م : حقيقي)؛

ويأتي بعد ذلك منطقة تعريف المتغيرات وهي أيضا مثل البرنامج، وتكون هذه المتغيرات محلية للاجراء فقط، أي أن البرنامج لا يعرفها خارج مداه، ويمكن تعريف نفس المتغير في اجراء آخر أو في البرنامج الرئيسي كمتغير مستغل تماما، ويجوز للاجراء استخدام متغيرات عامة معرفة خارجه.

برنامج صندوق؛

متغير

ل : صحيح؛

اجراء ارسم_صندوق (س١، ص١، س٢، ص٢، صحيح)؛
متغير
ل : صحيح؛
بداية
اذهب_ع_س (س١، ص١)؛
حيث ل := س١ **إلى** س٢ **اعمل** اكتب ('ـ')؛
حيث ل := ص١+١ **إلى** ص٢ **اعمل**
بداية
اذهب_ع_س (س١، ل)؛ اكتب ('|')؛
اذهب_ع_س (س٢، ل)؛ اكتب ('|')؛.
نهاية
اذهب_ع_س (س١، ص٢)؛
حيث ل := س١ **إلى** س٢ **اعمل** اكتب ('ـ')؛
نهاية، (٭ اجراء ارسم صندوق ٭)
بداية
امسح_شاشة؛
حيث ل := **١ إلى ١٠ اعمل**
ارسم_صندوق (ل×٢، ل×٢، ٣×ل، ٢×ل)؛
ارسم_صندوق (١، ١، ٨٠، ٢٣)؛
نهاية.

البرنامج السابق يقوم برسم صناديق مختلفة الاحجام بتمرير نقاط الصندوق الأربعة للاجراء، وحيث يستعمل طريقة القيم لأن البرنامج لا يريد تغيير قيمة المعطيات الأساسية كما في طريقة الاسناد.

مثال :
اجراء تبديل (**متغير** م، ن : صحيح)؛
متغير مؤقت : صحيح؛

بداية

م := ن؛

ن := مؤقت؛

نهاية؛

وينادى الاجراء من قبل البرنامج بالتعبير:

تبديل (ع،ك)؛

فيقوم بتبديل قيم المتغيرين (ع،ك) مع بعضهما، لكن إذا كانت المعطيات الأساسية في الاجراء معرفة كالآتي:

اجراء تبديل (م، ن : صحيح)؛

فلن يغير شيئا من المعطيات الفعلية (ع،ك) وسوف تبقى كما هي.

١٥ــ٢ــ١ الاجراءات القياسية

STANDARD PROCEDURES

يحتوي نظام اللغة على العديد من الاجراءات القياسية، بالإضافة إلى الاجراءات القياسية السابقة التي شرحناها في استعمالات السلاسل الرمزية والملفات والمؤشرات وعمليات الادخال والاخراج. الاجراءات التالية أيضا متوفرة في نظام اللغة.

امسح_سطر

تركيب الجملة : امسح_سطر؛

يمسح جميع الرموز من مكان المؤشر إلى نهاية السطر بدون تغيير مكان المؤشر.

امسح_شاشة

تركيب الجملة : امسح_شاشة؛

يمسح جميع الرموز التي على الشاشة، ويضع المؤشر في بداية السطر الأول.

هيء_شاشة

تركيب الجملة : هيء_شاشة

يرسل سلسلة المبدئية للوحدة الطرفية المعرفة في اجراء التشغيل الى الشاشة.

خروج_من_شاشة

تركيب الجملة : خروج_من_شاشة؛

يرسل سلسلة إعادة تهيئة الوحدة الطرفية المعرفة في اجزاء التشغيل الى الشاشة.

انتظر

تركيب الجملة : انتظر (وقت)؛

يعمل على انشاء دورة انتظار بعدد «وقت» ١/١٠٠٠ من الثانية، ويجب تعريف (وقت) كعدد صحيح.

احذف_سطر

تركيب الجملة : احذف_سطر؛

يحذف السطر الذي عليه المؤشر، وينقل جميع الأسطر التي تليه سطرا واحدا إلى الأعلى.

اضف_سطر

تركيب الجملة : اضف_سطر؛

يضيف سطرا فارغا في موقع المؤشر وينقل الأسطر التي تليه سطرا واحدا إلى أسفل.

اذهب ـع ـس

تركيب الجملة : اذهب ـع ـس (ع، س)؛

ينقل المؤشر إلى موقع العمود (ع) والسطر (س) وكلاهما عددان صحيحان ويكون المحور (١،١) هو بداية السطر الأول في الشاشة.

اخرج

تركيب الجملة : اخرج؛

يخرج من الكتلة الحالية، فعندما تستعمل داخل برنامج ثانوي فسوف تجعله يخرج منه، وإذا استعملت في البرنامج الرئيسي فسوف يتوقف تنفيذه.

قف

تركيب الجملة : قف؛

يتوقف تنفيذ البرنامج ويعود إلى نظام التشغيل الرئيسي.

أول ـعشوائي

تركيب الجملة : أول ـعشوائي؛

يجعل مولد الأرقام العشوائية يعطي قيمة مبدئية عشوائية.

انقل

تركيب الجملة : انقل (م١، م٢، ع)؛

ينقل هذا الاجراء كتلة كاملة من وحدات الذاكرة (بايت) وعددها (ع) مباشرة من الذاكرة الرئيسية، ابتدائا من الوحدة الأولى في الكتلة التي يشغرها المتغير (م١) إلى بداية أول وحدة في الكتلة التي يشغرها المتغير (م٢)، ويصنف (ع) بعدد صحيح، أما المتغيران (م١، م٢) فهما من أي صنف من الأصناف القياسية المعروفة في نظام اللغة.

املأ

تركيب الجملة : املأ (م، ع، ق)؛

يملأ مدى معين من الذاكرة بالقيمة (ق) وعدد وحدات ذاكرة (ع) ابتدائا من أول وحدة يحجزها المتغير (م)، ويعرف المتغير (م) من أي صنف، قياسي والعدد (ع) عدد صحيح، وتعرف القيمة (ق) إما بايت أو رمز.

١٥ــ٣ الدوال

FUNCTIONS

تعتبر الدوال كالاجراءات من حيث نوعيها، فهناك دوال قياسية في نظام اللغة (داخلية)، ودوال يصممها المبرمج بنفسه، وتستخدم الدوال عادة لحساب قيمة واحدة، بينما الاجراءات قد تعطي أكثر من نتيجة.

يتكون تعريف الدالة في البرنامج من صدارة الدالة، ومنطقة التعريف، يليها تعبيرات وجمل الدالة، وتتكون صدارة الدالة من الكلمة المحجوزة، «**دالة**» ثم اسم تعريفها، يليها المعطيات إذا وجدت، ثم صنف نتيجتها.

أمثلة:

دالة معدل (س، ص : حقيقي) : حقيقي؛

دالة احسب (**متغير** ق : صحيح) : حقيقي؛

تتشابه منطقة التعريف في الدالة والاجراء، أما تعبيرات الدالة فيجب أن تحتوي على جملة تخصيص قيمة معينة لاسم الدالة، واذا استعملت الدالة مرة اخرى في جملة التخصيص فسوف تقوم بعملية تكرار داخلي **(RECURSION)** لتنفيذ الدالة نفسها وتعتمد كل نتيجة على النتيجة التي قبلها وهذه الميزة ليست متوفرة إلا في بعض أنظمة التشغيل.

المثال التالي يوضح طريقة استعمال الدالة، فيقوم بأخذ عددين حقيقيين من المعطيات ثم يخصص قيمة معدلهما إلى الدالة.

مثال :

دالة معدل (س، ص : حقيقي) : حقيقي؛

بداية

معدل : = (س + ص) ÷ ٢,٠؛

نهاية؛

وعندما تكون نتيجة الدالة من صنف سلسلة فيجب تعريفها أولاً بمنطقة بمننطقة التصنيف في البرنامج الرئيسي فلا نعرف الدالة مثلا كالآتي:

دالة أحرف (سطر:صحيح) : **سلسلة** [٨٠]؛

فهذا غير مسموح في النظام، وإنما تكتب هكذا:

صنف

س : **سلسلة** [٨٠]؛

دالة احرف (سطر : صحيح) : س؛

١٥ــ٣ــ١ الدوال القياسية

STANDARD FUNCTIONS

الدوال الآتية يتم تطبيقها في نظام اللغة على الأنواع التالية:

١) السلاسل الرمزية

٢) الملفات وتطبيقاتها.

٣) المؤشرات.

وتنقسم الدوال من حيث تطبيقاتها إلى عدة فئات:

أ ــ الدوال الحسابية:

مطلق

تركيب الجملة : مطلق (ع)؛

تعمل على إرجاع القيمة المطلقة للعدد (ع)، ويعرف المعامل (ع) إما حقيقي أو صحيح، وتكون النتيجة من نفس نوع المعامل.

قظا

تركيب الجملة : قظا (ع)؛

تعطي الزاوية التي ظلها العدد (ع)، ويعرف المعامل (ع) اما حقيقي أو صحيح، والنتيجة دائما من نوع حقيقي.

جتا

تركيب الجملة : جتا (ع)؛

تعطي قيمة جيب التمام للعدد أو الزاوية (ع)، ويعرف المعامل (ع) اما حقيقي أو صحيح، والنتيجة من نوع حقيقي.

قوة

تركيب الجملة : قوة (ع)؛

تعمل على حساب الأس أو قوة العدد (ع)، ويعرف المعامل (ع) إما حقيقي أو صحيح والنتيجة من نوع حقيقي.

عشري

تركيب الجملة : عشري (ع)؛

تعطي الجزء العشري من العدد (ع)، وتحسب كالآتي:

عشري (ع) = ع − رقم_ص (ع)

ويعرف المعامل (ع) إما حقيقي أو صحيح، والنتيجة من نوع حقيقي

رقم_ص

تركيب الجملة : رقم_ص (ع)؛

تعطي الجزء الصحيح من العدد (ع)، ويعرف (ع)، إما حقيقي أو صحيح والنتيجة من نوع حقيقي.

لوغ

تركيب الجملة : لوغ (ع)؛

تعطي اللوغاريثم الطبيعي للعدد (ع)، ويعرف المعامل (ع) إما حقيقي أو صحيح، والنتيجة من نوع حقيقي.

جا

تركيب الجملة : جا (ع)؛

تعطي قيمة جيب العدد أو الزاوية (ع)، ويعرف المعامل (ع) إما حقيقي أو صحيح، والنتيجة من نوع حقيقي.

تربيع

تركيب الجملة : تربيع (ع)؛

تعطي تربيع العدد (ع) وهو حاصل ضرب العدد بنفسه (ع×ع)، ويعرف المعامل (ع) إما حقيقي أو صحيح، والنتيجة من نفس نوع (ع).

جذرت

تركيب الجملة : جذرت (ع)؛

تعطي الجذر التربيعي للعدد (ع)، ويعرف المعامل (ع) إما حقيقي أو صحيح، والنتيجة من نوع حقيقي.

ب ــ دوال المقاييس:

سابق

تركيب الجملة : سابق (م)؛

تعطي العنصر الذي يسبق عنصر المقياس (م) إذا كان موجودا، ويعرف المعامل (م) من أي نوع من المقاييس.

لاحق

تركيب الجملة : لاحق (م)؛

وهو عكس سابق فيعطي العنصر الذي يلحق عنصر المقياس (م) إذا كان موجودا، ويعرف (م) من أي نوع من المقاييس.

فردي

تركيب الجملة : فردي (ع)؛

تعطي نتيجة منطقية، وهي صحيح إذا كان العدد (ع) فردي، وخطأ إذا كان زوجي، ويعرف المعامل (ع) إما بعدد حقيقي أو صحيح.

ج ــ دوال التحويل

تسمى دوال التحويل.. لأنها تحول نوع قيم المقاييس إلى نوع آخر من المقاييس وهي كالآتي:

حرف

تركيب الجملة : حرف (ع)،

تعطي الحرف أو الرمز الذي يمثل العدد الصحيح (ع) في قائمة الشفرة (ASCII).

رقم

تركيب الجملة : رقم (م)؛

تعطي عددا صحيحا يمثل ترتيب العنصر (م) في المجموعة التي يعرف فيها المتغير (م).

قرب_كسر

تركيب الجملة : قرب_كسر (ع)؛

تعطي قيمة العدد الحقيقي (ع) مقرب إلى أقرب عدد صحيح، والنتيجة من نوع صحيح.

اهمل_كسر

تركيب الجملة : اهمل_كسر (ع)؛

تهمل الكسر العشري من العدد الحقيقي (ع)، والنتيجة عدد صحيح.

د ــ دوال قياسية متفرقة

اعلى

تركيب الجملة : أعلى (ع)؛

تعمل على أخذ الجزء العلوي من وحدة الذاكرة (بايت) في العدد الصحيح (ع) ووضعها في الجزء السفلي من البايت في نتيجة الدالة، ويكون الجزء العلوي من البايت صفراً، والنتيجة عددا صحيحا.

ادنى

تركيب الجملة : ادنى (ع)،

تعطي الجزء السفلي من وحدة الذاكرة (بايت) في العدد الصحيح (ع) ويكون الجزء االعلوي من البايت صفرا، والنتيجة عددا صحيحا.

ضغط_مفتاح

تركيب الجملة : ضغط_مفتاح؛

تعطي نتيجة منطقية، وهي صحيحة إذا تم الضغط على أحد مفاتيح اللوحة، وخطأ إذا لم يحدث ذلك.

عشوائي

تركيب الجملة : ١) عشوائي؛

٢) عشوائي (ع)؛

التركيب الأول يعطي العدد العشوائي الذي أعلى من ــ أو يساوي صفر ــ وأقل من واحد، والنتيجة عدد حقيقي، أما التركيب الثاني فيعطي العدد العشوائي الذي أعلى من ــ أو يساوي صفر ــ وأقل من العدد الصحيح (ع) والنتيجة عدد صحيح.

عدد_معطيات

تركيب الجملة : عدد_معطيات؛

تعطي عددا صحيحا يمثل عدد المعطيات التي مررت عن طريق نظام التشغيل الرئيسي (DOS) إلى البرنامج في ذاكرة الأوامر المؤقتة (BUFFER)، ويعامل رمز الفراغ «SPACE» ورمز الجدولة «TAB» كفواصل بين المعطيات.

معطيات_سلسلة

تركيب الجملة : معطيات_سلسلة (س)؛

تعطي المعطى الذي ترتيبه (س) في سلسلة المعطيات في ذاكرة الأوامر المؤقتة (BUFFER)، والنتيجة من نوع سلسلة رموز.

حجم_ملف

تركيب الجملة : حجم_ملف (اسم)؛

تعطي عدد وحدات الذاكرة التي يحتجزها المتغير (اسم)، والنتيجة عدد صحيح.

بدل

تركيب الجملة : بدل (ع)؛

تقوم بإبدال الجزء العلوي والسفلي من بايت العدد الصحيح (ع) مع بعضهما، والنتيجة عدد صحيح.

١٥ـ٤ التعريف الأمامي

FORWARD DECLARATION

كما ذكر سابقا لا بد من تعريف البرامج الفرعية من الاجراءات والدوال قبل استعمالها في كتلة البرنامج، فإذا أسند للبرنامج الفرعي قبل تعريفه يكون غير معروف في المترجم، ويؤدي الى حدوث أخطاء عند التنفيذ في التركيب، وللمترجم طريقته الخاصة بمعرفة البرامج الفرعية، فعند التنفيذ يقوم بقراءة البرنامج من بدايته، ويقرأ الثوابت والمصنفات والمتغيرات وكذلك الاجراءات والدوال، ويقوم بوضع جدول خاص يضعها جميعا فيه.

فعند أستعمال البرنامج الفرعي، يقرأ المترجم الجدول، فإذا كان اسمه موجودا.. أكمل التنفيذ، أما اذا كان غير موجود فسيتوقف التنفيذ. وقد يحتاج المبرمج الى اسناد البرنامج الفرعي قبل تعريفه، ولهذا أتيح له استخدام التعريف الأمامي حيث يقوم المترجم بوضع مقدمة البرنامج الفرعي في الجدول الخاص. ويستعمل التعريف الأمامي، لتعريف مقدمة البرنامج الفرعي « **SUB PROGRAM**» قبل تصميم الهيكل الداخلي للبرنامج، وتكون المقدمة معرفة فقط مع المعطيات اللازمة، وتعرف الاجراءات والدوال الأمامية في آخر منطقة تعريف المتغيرات حيث يتكون من مقدمة البرنامج الفرعي مع المعطيات اللازمة يليها الكلمة المحجوزة «**امامي**»، اما هيكل البرنامج الفرعي فيكتب فيما بعد في موقعه الأساسي، وتعرف مقدمته بدون معطيات.

وتأتي صيغة التعريف الأمامي في البرنامج كالآتي:

```
**برنامج** مثال_الأمامي؛
{منطقة تعريف المتغيرات}
**اجراء** س_ص (س، ص : صحيح)، **امامي**؛
 **اجراء** م_ن (**متغير** د : حقيقي)؛
  **بداية**
    {محتويات الاجراء}
     س_ص؛
  **نهاية**؛

  **اجراء** س_ص؛
  **بداية**
    {محتويات الاجراء}
  **نهاية**؛
**بداية** {البرنامج الرئيسي}
 .
 .
 .
   {جمل تنفيذية}
**نهاية**.
```

```
**برنامج** تجربة_الأمامي؛
 **متغير**
 س، ص : صحيح؛
 **اجراء** مثال_١ (ص : صحيح)؛ **امامي**؛
```

```
اجراء مثال_٢ (ص : صحيح)؛
بداية
  اكتب_سطر (ص)؛
  ص := ص + ١ ؛
  مثال_١ (ص)؛
نهاية؛ {الاجراء مثال ٢}
اجراء مثال_١؛
 بداية
    ص := ص + ١ ؛
    اكتبسطر (ص)؛
نهاية؛ {الاجراء مثال ١}
بداية {البرنامج الرئيسي}
  س:= ١؛
  مثال_٢ (س)؛
نهاية.
```

```
برنامج اجراءات_ودوال_قياسية؛
اجراء روتين_مكرر؛
بداية
    اذهب_ع_س (٣٠، ٢٣)؛
    لون_نص (٥+غمز)؛
    اكتبسطر (' اضغط مفتاح العودة «ENTER» للاستمرار')؛
    اقرأ؛
    لون_نص (اصفر)؛
    امسح_شاشة؛
    اكتبسطر؛
نهاية؛
```

(٭ الاجراء يعطي القيمة المطلقة للاعداد من -٥ الى ٥ ثم يحسب قوتها كما يعطي قيمة الجزء العلوي والسفلي من بايت العدد الصحيح ٭)

اجراء مطلق_وقوة_عدد؛

متغير

س، ص : صحيح؛

ن : حقيقي؛

بداية

اكتبسطر ('س مطلق (س) قوة (س) اعلى (س) ادنى (س)')

حيث س := -٥ الى ٥ اعمل

اكتبسطر (' '،س،' '،مطلق (س) :٢،

' '، قوة (س) :٧:٢،' '، اعلى (س)'، ' '، ادنى (س) :٥)؛

نهاية؛

(٭ يعمل على اعطاء بعض البيانات اللازمة عن الزوايا من ٠ الى ٣٦٠ ٭)

اجراء اعمل_جاجتاقظا؛

متغير زاوية : صحيح؛

دالة قطري (زاوية : صحيح) : حقيقي؛

بداية

قطري := مطلق (زاوية **باقي** ٣٦٠) × **طه** ÷ ١٨٠,٠

نهاية؛

بداية

زاوية := ٠؛

اكتبسطر ('الزاوية جا جتا قظا')؛

طالما (زاوية > = ٣٦٠) اعمل

بداية

اكتب (' '،زاوية : ٣ ، ' '، جا (قطري (زاوية)):٨:٣،

' '، جتا (قطري (زاوية)):٨:٣)؛

```
    اكتبسطر ('    ' قطا (قطري (زاوية) ):٨:٣)؛
      زاوية := زاوية + ٣٠
   نهاية {طالما}
نهاية ؛
(٥ يعطي العدد العشري والصحيح من العدد الحقيقي ح ثم تربيعه وجذره     ٥)
اجراء_عدد_عشري؛
متغير   ح : حقيقي؛ ص : صحيح؛
بداية
  اكتبسطر (' ح   عشري (ح)   صحيح (ح)   تربيع (ح)   جذرت(ح) ')؛
  حيث ص := ١ الى ١٠ اعمل
   بداية
     ح := عشوائي + عشوائي × ١٠٠؛
     اكتب (ح:٨:٤ ، '    ' ، عشري (ح) :٧ :٣ ، '    ' ،
       رقمص (ح) :٧:٣ ، ' '،تربيع (ح) :٧   :٣)؛
     اكتبسطر ('   ' ، جذرت (ح) :٧:٣)؛
   نهاية
نهاية ؛
بداية
  امسح_شاشة؛
  مطلق_وقوة_عدد؛
  روتين_مكرر؛
  اعمل_جاجتاقطا؛
  روتين_عشري؛
  عدد_عشري؛
نهاية.
```

الباب السادس عشر

الخواص الإضافية الخاصة بالحاسب الشخصي

EXTRA FEATURS OF THE IBM PC

يتميز نظام التشغيل الرئيسي «MS-DOS 3.X» أو «PC-DOS 3.X» بوجود أوامر متنوعة للتحكم بالشاشة والسماعة ولوحة المفاتيح وجميع الوحدات الطرفية في الجهاز، ولقد روعي في نظام نبراس هذه المميزات الخاصة بالجهاز الشخصي وكيفية الاستفادة منها، فمثلا التحكم التام بالنهاية الطرفية «الشاشة» كالتحكم بألوان الاخراج، ومسح الشاشة وكذلك اصدار الصوت من السماعة، ومميزات أخرى عديدة كانت موضع الاعتبار سوف نوضحها في هذا الباب.

١٦ــ١ التحكم في وضع الشاشة

SCREEN MODE CONTROL

يتوفر في الأسواق أنواع عديدة من الوحدات الطرفية «الشاشة»، كما أن كل نوع له مميزات خاصة من حيث الأطوال والألوان، ويوجد حالتان تعمل عليها أجهزة الحاسب الشخصي، الحالة الأولى ــ «**حالة النص**» حيث الشاشة تظهر ٢٥٦ رمزا محتملا من رموز الشفرة على الشاشة، والحالة الثانية ــ «**حالة الرسم**» حيث يستطيع المبرمج التحكم بكل **جزيء** (PIXEL) موجود على الشاشة حيث يمكن رسم خطوط ومربعات ودوائر وأشكال كثيرة بدون تحديد.

ويعرف «**الجزيء**» على أنه أصغر نقطة ضوء موجودة على الشاشة، وتقاس

جودة الشاشة بهذه الوحدة، كما أن الرموز تتركب من هذه الجزيئات حيث يستطيع المبرمج أن يتحكم في الجزيء الذي يريده مضاء والجزيء الذي يريده مظلم. سوف نتحدث عن بعض أشكال الشاشة التي تتوفر في كلا الحالتين حيث أنه روعي في نظام نبراس أغلب الحالات والأشكال للنهاية الطرفية «الشاشة».

والحالات المتوفر في نظام نبراس يبينها الجدول التالي:

الحالة	شكل الشاشة
ـ النص	(٢٥ سطر × ٤٠) أو ٨٠ حرف
ـ لون الرسم	(٣٢٠ × ٢٠٠) جزيء ملون
ـ الرسم	(٣٢٠ × ٢٠٠) جزيء أسود وأبيض
ـ الرسم العالي	(٦٤٠ × ٢٠٠) جزيء أسود + لون واحد فقط

جدول ١٦ـ١ حالات الشاشة

١٦ـ١ـ١ حالة النص

TEXT MODE

عند كتابة البرامج وتحرير النصوص فإن الشاشة تكون في حالة تسمى «**حالة النص**» ، وتكون في هذه الحالة أطوال الشاشة اما ٢٥ سطر × (٤٠ أو ٨٠) عمود أي ما مجموعة ٢٠٠٠ رمز في الشاشة الواحدة، حيث أن الحاسب الشخصي يستطيع عرض أي رمز من رموز الشفرة القياسية، فأشكالها

مخزنة في ذاكرة الجهاز دائما، ولاظهار الرمز فإن الحاسب الشخصي يستخدم **بطاقة إعداد**، وهي عبارة عن لوحة من دوائر كهربائية توصل الجهاز مع الشاشة.

يمكن تغيير شكل الشاشة إلى عدة أشكال، باستعمال الاجراء الداخلي «**حالة_نص**» للانتقال من شكل إلى آخر وهو أحد الاجراءات الداخلية الموجودة في نظام اللغة، وهو خاص بشكل الشاشة فقط، ولهذا الاجراء خمسة حالات وهي:

١) حالة_نص؛
٢) حالة_نص (٠)؛
٣) حالة_نص (١)؛
٤) حالة_نص (٢)؛
٥) حالة_نص (٣)؛

في الحالة الأولى، يكتب الاجراء بدون معطيات، فيضع الشاشة إلى آخر حالة كانت فيها، أو الوضع الحالي، وعند اعطاء معطيات كما في الاحتمالات الأخرى، فإن الشاشة تتغير إلى هذا الشكل بدون أخذ أي أعتبار للشكل الحالي للشاشة كالآتي:

حالة_نص (٠) يتغير شكل الشاشة إلى أسود وأبيض، وتصبح سعة السطر في الشاشة ٤٠ رمزا فقط.

حالة_نص (١) يتغير شكل الشاشة إلى حالة ملونة، وتصبح سعة السطر في الشاشة ٤٠ رمزا فقط.

حالة_نص (٢) يتغير شكل الشاشة إلى أسود وأبيض، وتصبح سعة الشاشة ٨٠ رمزا.

حالة_نص (٣) يتغير شكل الشاشة إلى حالة ملونة، وتصبح سعة الشاشة ٨٠ رمزا.

يجب على المبرمج استعمال الاجراء (حالة نص) قبل الخروج من البرنامج عند استعماله لأي حالة من الحالات الأربع السابقة، وذلك لإعادة الشاشة إلى الحالة الطبيعية.

١٦ـ١ـ١ـ١ الألوان الخاصة بحالة النص

TEXT MODE COLORS

عند استعمال حالة النص، يمكن تغيير شكل الشاشة إلى مجموعة من الألوان حسب رغبة المبرمج.. فيمكن اختيار لون لكل حرف أو رمز عند الاخراج يختلف عن الآخر، كما يمكن أيضا التحكم في اللون الخلفي لكل رمز والمقصود باللون الخلفي.. أي لون الفراغ الذي خلف الرمز في الشاشة، فالحاسب الآلي يخزن بيانات الرموز والألوان في منطقة معينة من الذاكرة تسمى «**ذاكرة العرض**» وهي التي تخزن البيانات التي تخبر الحاسب الآلي عن الرموز والألوان التي سيعرضها على الشاشة مع إن هذه الذاكرة موجودة على بطاقة إعداد الفيديو، الا إنها تعتبر جزء من الذاكرة الرئيسية.

يتوفر في نظام اللغة. إجراءان قياسيان يستعملان للتحكم بالألوان فيمكن استعمال ستة عشر لونا في الاجراء «**لون نص**»، وثمانية ألوان في الاجراء «**خلف_نص**»، والاختيار اللون المفضل، يستعمل رقم أو اسم اللون كمعطيات للاجراء حسب الجدول التالي:

ألوان «خلف نص»	ألوان «لون نص»
(٠) أسود	(٨) رمادي غامق
(١) أزرق	(٩) أزرق فاتح
(٢) أخضر	(١٠) أخضر فاتح
(٣) سماوي	(١١) سماوي فاتح
(٤) أحمر	(١٢) أحمر فاتح
(٥) وردي	(١٣) وردي فاتح
(٦) بني	(١٤) أصفر
(٧) رمادي فاتح	(١٥) أبيض
	بالإضافة إلى جميع ألوان خلف نص

جدول ١٦ـ٢ الألوان الخاصة بحالة النص

لون_نص

تركيب الجملة : لون_نص (لون)؛

هذا الاجراء يغير لون أي نص سواء كان رمز أو كلمة أو نص إلى اللون المطلوب و (لون) يمكن أن يكون أحد الألوان المدونة في الجدول السابق أو أحد الأرقام من صفر إلى ١٥. ويمكن جعل اللون يغمز بزيادة الرقم ١٦ إلى أرقام الألوان أو باستعمال الكلمة المحجوزة «**غمز**».

أمثلة :

لون_نص (٣)؛ لاختيار اللون السماوي.
لون_نص (سماوي)؛
لون_نص (اصفر)؛

ويمكن جعل النص يؤشر أو يغمز بزيادة الرقم ١٦ إلى ٣ كما سبق ذكره أو زيادة الكلمة «غمز» كما في المثال الآتي:

لون_نص (١٩)؛ لاختيار اللون السماوي مع الغمز (٣+١٦) أو لون_نص (سماوي +غمز)؛

(خلف_نص)

تركيب الجملة : خلف_نص (لون)؛

هذا الاجراء يغير اللون الخلفي للنص سواء كان رمزاً أو كلمة أو نص إلى اللون المطلوب و (لون) يمكن أن يكون أحد الألوان الثمانية المدونة في الجدول السابق أو أحد الأرقام من صفر إلى ٧.

أمثلة :

خلف_نص (٤)؛ لاختيار اللون الأحمر كلون خلفي

خلف_نص (احمر)؛

١٦ـ١ـ١ـ٢ محاور الشاشة

SCREEN COORDINATES

قد يحتاج المبرمج في بعض الحالات للذهاب إلى مكان معين على الشاشة أو لمعرفة السطر والعمود الذي يقع عليه المؤشر في الشاشة. تحتوي الشاشة على محاور كالخريطة الجغرافية، يرمز الحرف «**س**» غالبا للمحور الأفقي وكذلك يرمز الحرف «ع» للمحور العمودي وتعتبر الزاوية العلوية اليمنى هي المحور (١، ١) بينما المحور الذي في أسفل الزاوية اليسرى هو المحور (٨٠، ٢٥).

اذهب_ع_س

تركيب الجملة : اذهب_ع_س (ع، س)؛

يمكن تحريك المؤشر إلى أي مكان على الشاشة بواسطة هذا الاجراء حيث يعطي المحور المراد وضع المؤشر عليه (ع، س) كما يجب أن لا يتعدي نطاق محاور الشاشة.

اين_س

تركيب الجملة : اين_س؛

نتيجة الدالة عدد صحيح. ويمثل العدد رقم السطر (س) الذي يقع عليه المؤشر.

اين_ع

تركيب الجملة : اين_ع؛

نتيجة الدالة عدد صحيح. ويمثل العدد رقم العمود (ع) الذي يقع عليه المؤشر.

١٦ــ١ــ٢ حالات الرسم

GRAPHICS MODES

هذه الحالة تختلف عن حالة النص في أطوال الشاشة اما (٣٢٠ × ٢٠٠) أو (٦٤٠ × ٤٠٠) جزيء حسب الحالة المختارة، ويمكن رسم أي شكل على الشاشة حيث يمكن التحكم بالاجزاء الموجودة على مساحة الشاشة.

ان اجهزة «IBM» الشخصية تحتوي على عشرة حالات رسم تقريبا ولكن نظام نبراس يحتوي فقط على ثلاث منها، ويمكن الانتقال من حالة الى أخرى من بين الثلاث حالات وهي:

— **لون_رسم** (٣٢٠ × ٢٠٠) جزيء ملون

— **حالة_رسم** (٣٢٠ × ٢٠٠) جزيء أسود وأبيض

— **رسم_عالي** (٦٤٠ × ٢٠٠) جزيء أسود + لون واحد فقط

ولا بد من استعمال الاجراء «**حالة_نص**» قبل الخروج من البرنامج بعد استعمال أي حالة من الحالات الثلاث السابقة للرجوع إلى حالة النص.

١٦ـ١ـ٢ـ١ حالة لون الرسم

GRAPH COLOR MODE

للانتقال إلى هذه الحالة يستعمل الاجراء «**لون_رسم**» ، ولا يمكن استعمال الألوان كما هي في حالة النص.. إذ أمها مقسمة إلى أربع مجموعات من الألوان، وكل مجموعة تحتوي على ثلاثة ألوان مع اللون الرابع كلون خلفي للشاشة. ويستعمل الاجراء «**مجموعة_الوان**»، لنحصل على مجموعة الألوان أولا.. ثم الحصول على أحد الألوان الثلاثة المتوفرة في هذه المجموعة ويستعمل الاجراء «**نقط**» ، لعمل نقاط من هذا اللون.

لون_رسم

تركيب الجملة : **لون_رسم؛**

هذا الاجراء الداخلي يغير وضع الشاشة إلى حالة الرسم بالألوان، مع تغيير أطوال الشاشة إلى (٣٢٠ × ٢٠٠) ليصبح طول السطر من صفر إلى ٣١٩ وطول العمود من صفر إلى ١٩٩ جزيء.

مجموعة_الوان

تركيب الجملة : مجموعة_ألوان (س)؛

يستعمل الاجراء في حالة «لون الرسم» للحصول على إحدى المجموعات

الأربعة الخاصة بهذه الحالة. يمثل المتغير «س» أحد الأرقام من صفر إلى ٣، وكل رقم يشكل مجموعة من ثلاثة ألوان كما هو مبين بالجدول:

رقم اللون	٠	١	٢	٣
مجموعة الوان (٠)	لون خلفي	أخضر	أحمر	بني
مجموعة الوان (١)	لون خلفي	سماوي	وردي	رمادي فاتح
مجموعة الوان (٢)	لون خلفي	أخضر فاتح	أحمر فاتح	أصفر
مجموعة الوان (٣)	لون خلفي	سماوي فاتح	وردي فاتح	أبيض

جدول ١٦ـ٣ مجموعة الألوان الخاصة في حالة الرسم

```
برنامج تجربة ألوان؛
بداية
    لون_رسم؛
    مجموعة_الوان (٢)؛
    نقط (١٠٠، ٢٠٠، ٣)؛ (* سوف يقوم بالتنقيط باللون الأصفر *)
نهاية.
```

عند تنفيذ البرنامج السابق، نلاحظ أن الشاشة قد تحولت إلى حالة الرسم بواسطة الاجراء «لون رسم» وقد تم اختيار مجموعة الألوان الثانية للرسم، أما الاجراء «نقط»، يضع نقطة واحدة على الشاشة بلون أصفر، فهو اللون الثالث من المجموعة الثانية.

وعند النظر إلى الجدول السابق، نلاحظ أن المجموعة الثانية تتكون من

الألوان أخضر فاتح، أحمر فاتح، أصفر، ويتم تغيير اللون بالمجموعة بتغيير الرقم الأخير من معطيات الاجراء (نقط). فعند استعمال البرنامج السابق مع «نقط (١٠٠، ٢٠٠، ١)» نلاحظ أن جميع الألوان على الشاشة تتغير إلى اللون الأخضر الفاتح. ولتغيير اللون إلى أي لون آخر.. يجب أولا تحويل الحالة إلى حالة «**لون الرسم**» ثم اختيار المجموعة التي ينتمي إليها اللون المطلوب، ويستخدم الاجراء «**مجموعة_الوان**»مع إضافة الرقم الذي يمثل اللون في المجموعة، ثم يوضع اللون المطلوب في الاجراء «**نقط**» ويصبح البرنامج السابق:

برنامج تجربة_ألوان،

بداية

لون_رسم؛

مجموعة_الوان (٠)؛

نقط (١٠٠،٢٠٠،٢)؛

نهاية

١٦ــ١ــ٢ــ٢ حالة الرسم

GRAPH MODE

هذه الحالة مشابهة كثيرا لحالة لون الرسم لكنها تختلف في استخدام الألوان حيث أنها تستعمل ألوانا أقل، فهي مصممة للشاشات التي تعرض اللون الأخضر على اللون الأسود ولهذا لا تستفيد من الألوان المتوفرة، والألوان الرئيسية في هذه الحالة هي الأزرق والأحمر والرمادي الفاتح. وتستخدم مجموعتين من مجموعات الألوان، فمجموعة (٠) تعرض هذه الألوان الغامقة من الألوان الرئيسية ومجموعة (١) تعرض الألوان الفاتحة. وللانتقال إلى هذه الحالة يستعمل الاجراء «**حالة_رسم**».

حالة_رسم

تركيب الجملة : حالة_رسم؛

يغير وضع الشاشة إلى حالة الرسم بالأسود والأبيض وتغير أطوال الشاشة إلى (٣٢٠ × ٢٠٠) جزيء لتصبح أطوال السطر من صفر إلى ٣١٩ جزيء وطول العمود من صفر إلى ١٩٩ جزيء

لون_رسم_خلفي

تركيب الجملة : لون_رسم_خلفي (لون)؛

يغير اللون الخلفي للشاشة إلى اللون المطلوب والمتغير (لون) يمكن أن يكون أحد الألوان المدونة في الجدول التالي أو أحد الأرقام من صفر إلى ١٥.

الألوان الغامقة	الألوان الفاتحة
(٠) أسود	(٨) رمادي غامق
(١) أزرق	(٩) أزرق فاتح
(٢) أخضر	(١٠) أخضر فاتح
(٣) سماوي	(١١) سماوي فاتح
(٤) أحمر	(١٢) أحمر فاتح
(٥) وردي	(١٣) وردي فاتح
(٦) بني	(١٤) أصفر
(٧) رمادي فاتح	(١٥) أبيض

جدول ١٦ـ٤ الألوان الخلفية الخاصة بالرسم

١٦ـ١ـ٢ـ٣ حالة الرسم العالي

HIGH RESOLUTION MODE

تسمى بحالة الرسم العالي لأنها تكون أكثر دقة وأعلى نوعية للرسم. مع ان هذه الحالة أكثر دقة الا أنها أقل ألوانا من الحالات الأخرى، حيث يعرض الجهاز هذه الحالة بلون واحد فقط (أي لون من ١٦ لون) ويبقى اللون الخلفي للشاشة أسود بدون تغيير، وللانتقال إلى هذه الحالة يستعمل الاجراء «**رسم_عالي**» ، ويتم اختيار لون الرسم باستعمال الاجراء «**لون_رسم_عالي**».

رسم_عالي

تركيب الجملة : رسم_عالي؛

يغير وضع الشاشة إلى حالة الرسم العالي، وتتغير أطوال الشاشة إلى (٦٤٠ × ٢٠٠) جزيء، حيث يصبح طول السطر من صفر إلى ٦٣٩ جزيء وطول العمود من صفر إلى ١٩٩ جزيء.

لون_رسم_عالي

تركيب الجملة : لون_رسم_عالي (لون)؛

يغير لون الرسم إلى اللون المطلوب، ويغير اللون بتغيير (لون) حيث لون عبارة عن عدد صحيح من صفر إلى ١٥.

ويمكن اختيار أحد الألوان من الجدول التالي:

الألوان الغامقة	الألوان الفاتحة
(٠) أسود	(٨) رمادي غامق
(١) أزرق	(٩) أزرق فاتح
(٢) أخضر	(١٠) أخضر فاتح
(٣) سماوي	(١١) سماوي فاتح
(٤) أحمر	(١٢) أحمر فاتح
(٥) وردي	(١٣) وردي فاتح
(٦) بني	(١٤) أصفر
(٧) رمادي فاتح	(١٥) أبيض

جدول ١٦ـ٥ الألوان الخاصة بحالة الرسم العالي

١٦ـ٢ النافذة

WINDOW

النافذة : منطقة من الشاشة تعرف على أنها الشاشة الأصلية وتظهر رموز ورسوم الاخراج في هذه النافذة، وهناك نوعان من النافذة «**نافذة النص**» وتستخدم في حالة النص، و «**نافذة الرسم**» وتستخدم في حالة الرسم.

١٦ـ٢ـ١ نافذة النص

TEXT WINDOW

عندما تكون الشاشة في حالة النص يمكن تعليم منطقة من الشاشة لتصبح نافذة نص، ويستعمل الاجراء «**نافذة**» مع المعطيات اللازمة لعمل تلك النافذة.

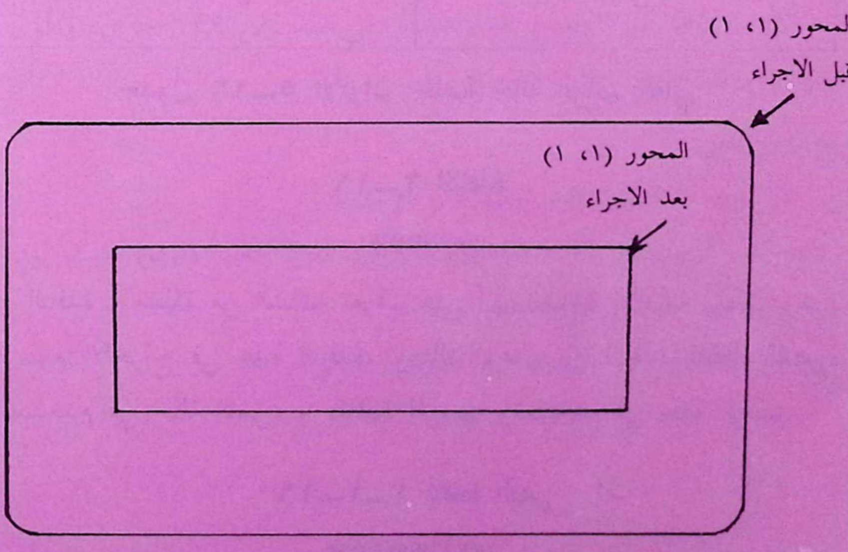
نافذة

تركيب الجملة : نافذة (س١، ص١، س٢، ص٢)؛

المعطيات (س١، ص١، س٢، ص٢) تمثل نقاط الشاشة التي تبدأ وتنتهي منها النافذة، ويمثل المعطيات (س١،ص١) محور الزاوية العلوية اليمنى في الشاشة، والمعطيات (س٢، ص٢) محور الزاوية اليسرى من أسفل الشاشة، في هذه الحالة يصبح المحور (س١، ص١) هو المحور (١،١) على الشاشة ولا يمكن تعريف نافذة أطوالها أقل من (٢ × ٢).

مثال:

اذا افترضنا أن الشكل التالي يمثل الشاشة، فإن الاجراء نافذة (١٠، ١٢، ٦٠، ٢٠)؛ يجعل الشاشة الفعلية:



شكل ١٦ــ١ نافذة النص.

فإن الشاشة تصبح المتسطيل الداخلي والمحور (١،١) يصبح الزاوية اليمنى

من أعلى الشكل الداخلي. علما بإن الحالة النظامية للنافذة هي (١، ١، ٨٠، ٢٥) أي الشاشة بأكملها.

١٦ـ٢ـ٢ نافذة الرسم

GRAPHICS WINDOW

عندما تكون الشاشة في حالة الرسم يمكن تعليم منطقة من الشاشة لتصبح نافذة رسم، ويستعمل الاجراء «**نافذة_رسم**» مع المعطيات اللازمة لعمل تلك النافذة. ولا تختلف نافذة الرسم عن نافذة النص من حيث المعطيات، ولكن وجه الاختلاف يأتي في أطوال تلك المعطيات.

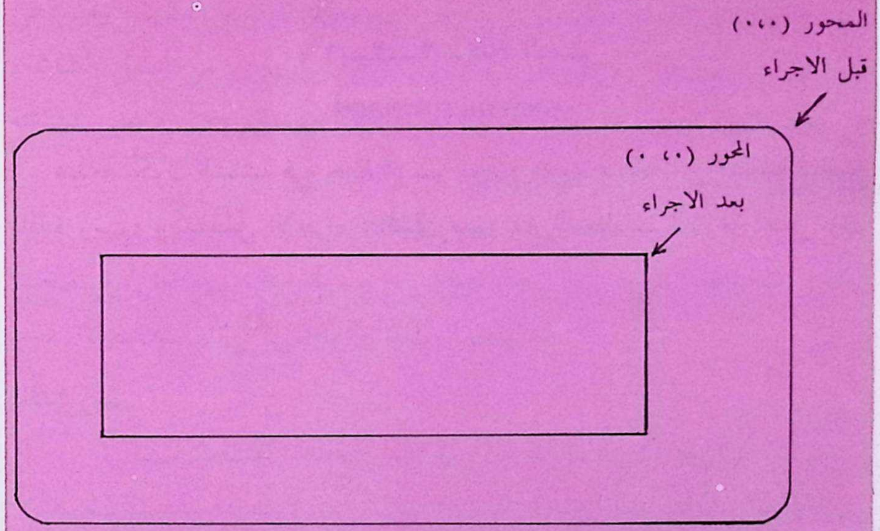
نافذة_رسم

تركيب الجملة : نافذة_رسم (س١، ص١، س٢، ص٢)؛

المعطيات (س١، ص١، س٢، ص٢) تمثل محاور الشاشة التي تبدأ وتنتهي فيها النافذة، ويمثل المحور (س١، ص١) الزاوية اليسرى من أعلى الشاشة، والمحور (س٢، ص٢) يمثلان الزاوية اليمنى من أسفل الشاشة، ويصبح المحور (س١، ص١) هو محور الأصل (٠،٠) وجميع الرسومات أو النصوص الموجودة خارج نطاق النافذة سوف تختفي وتبقى الأشياء التي داخل النافذة.

مثال:

إذا افترضنا أن الشكل التالي يمثل الشاشة، فالاجراء نافذة (٦٠، ١٢٠، ٢٣٠، ٢٠٠)؛ يجعل الشاشة الفعلية كالآتي:



شكل ١٦ــ٢ نافذة الرسم

فتصبح الشاشة المستطيل الداخلي، والمحور (٠،٠) يصبح في الزاوية اليمنى من أعلى الشكل الداخلي. علما بأن الحالة النظامية للنافذة هي (**٠،٠،٣١٩،١٩٩**) في «حالة رسم» و (**٠،٠،٦٣٩،١٩٩**) في حالة «رسم ـ عالي».

في جميع حالات الرسم يستعمل الاجراء «**نقط**» لعمل نقطة في أي مكان في الشاشة مع اللون المطلوب، ويستعمل الاجراء «**ارسم**» لرسم خط مستقيم بين نقطتين في الشاشة مع اللون المطلوب وتتم عملية التحكم في الألوان والمكان باعطاء الاجرائين المعطيات المطلوبة.

نقط

تركيب الجملة : نقط «س، ص، لون)؛

يضع نقطة في المكان المحدد بواسطة المحور (س،ص)، مع اللون المطلوب (لون). ويجب أن يكون المتغير (لون) عدد صحيح.

مثال:

نقط (٢٦٥، ١٦٥، ٢)؛

المثال يضع نقطة حمراء في المحور (٢٦٥، ١٦٥).

ارسم

تركيب الجملة : ارسم (س١،ص١، س٢، ص٢، لون)؛

يرسم خط بين محورين في الشاشة أو النافذة بواسطة المعطيات (س١،ص١، ص٢، ص٢) حيث المحور (س١، ص١) يمثل النقطة الأولى والمحور (س٢، ص٢) يمثل النقطة الثانية. والمتغير (لون) عدد من نوع صحيح.

مثال:

ارسم (٥٠، ١٠٠، ٢٠٠، ١٨٠، ١)؛

المثال يرسم خط باللون الأزرق بين المحورين (٥٠،١٠٠)، (٢٠٠، ١٨٠).

١٦ـ٣ الصوت

SOUND

يوجد في معظم أجهزة الحاسب الآلي الشخصية سماعة صغيرة، لاصدار أصوات إما موسيقية، أو التنبيه لحصول خطأ في استعمال بعض البرامج، وتعمل هذه السماعة باستعمال الاجراء «**صوت**» .

صوت

تركيب الجملة : صوت (س)؛

(س) عبارة عن عدد صحيح يوضح الذبذبة المطلوبة، فيمكن عمل نغمة معينة أو موسيقية بواسطة التحكم في مقدار الذبذبة (س)، ويجب استعمال الاجراء «**لاصوت**» بعد استعمال الاجراء «**صوت**».. وكلا الاجراءان يمثلان مفتاح السماعة كمفتاح اشعال المصباح الكهربائي إذ يجب اقفال مفتاح الاشعال لاطفاء الضوء، كما يجب اقفال مفتاح السماعة لإلغاء الصوت.

الصوت.

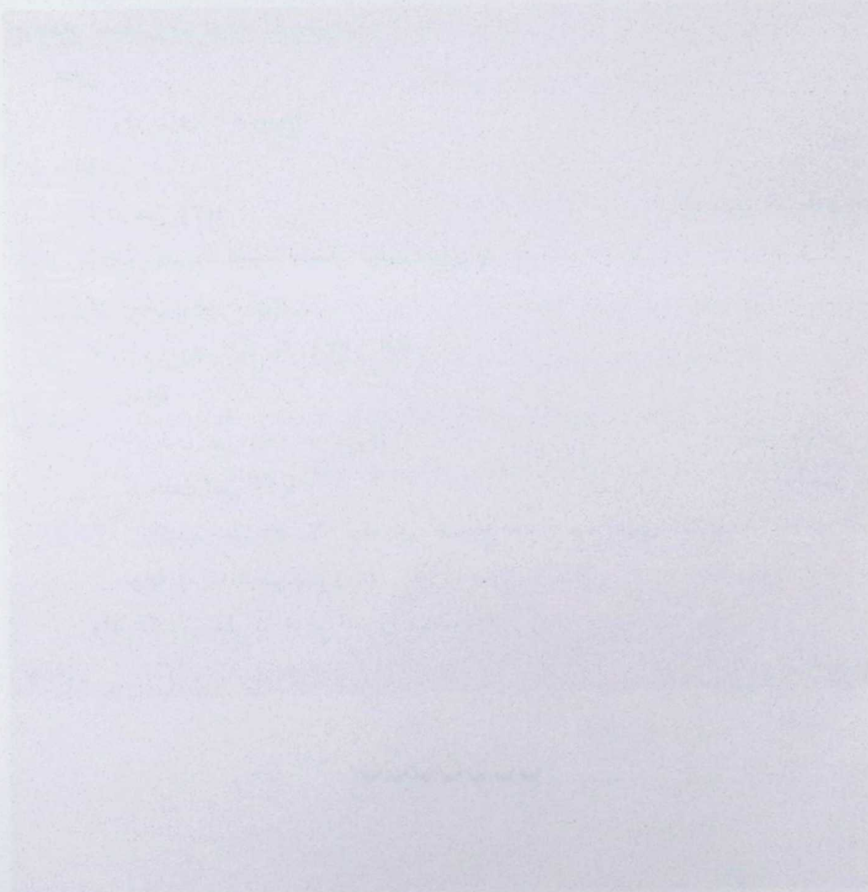
مثال:

```
برنامج صوت لاصوات؛
بداية
    صوت (٥٠٠)؛
    انتظر (١٠٠٠)؛
    لاصوت؛
نهاية.
```

المثال السابق يجعل السماعة تصدر صوت بذبذبة مقدارها ٥٠٠ هيرتز «Hz 500» لمدة ثانية واحدة.

كما يمكن جعل السماعة تصدر صوت من نغمة واحدة بكتابة =٧.

```
برنامج اختبار_الرقم_٧_لاصدار_صوت؛
متغير
    رقم_سري : صحيح؛
بداية
    لون_نص (٣)؛                                          (* اللون السماوي *)
    اكتب_سطر (' الرجاء ادخال الرقم السري')؛
    اقرأ_سطر (رقم_سري)؛
    اذا رقم_سري <> ١٢٣٤ اذن
        بداية
            لون_نص (١٥ + غمز)؛                          (* ابيض ويؤشر *)
            خلف_نص (٤)؛                                  (* أحمر *)
            اكتب_سطر (#٧ ،'الرقم غير صحيح حاول مرة اخرى !!!!')؛
        نهاية (* اذا الرقم السري *)
    والا اكتب_سطر (' الرقم السري صحيح')؛
نهاية.                                                    (* البرنامج *)
```



الباب السابع عشر

الخواص الرئيسية الخاصة بنظام التشغيل الرئيسي

THE MAIN FEATURES PC- DOS AND MS-DOS

يحتوي نظام التشغيل «DOS» على أوامر متنوعة ومفيدة، وقد يحتاج المبرمج إلى تنفيذ بعض الأوامر داخل البرنامج، ولذلك صمم في نظام **نبراس** عدة اجراءات ودوال لتساعد المبرمج من الدخول بنظام التشغيل «DOS» والتحكم التام بالملفات والذاكرة أو الدليل بدون الخروج من نبراس. هذا بالإضافة إلى خواص أخرى، سوف نوضحها في هذا الباب.

١٧ــ١ الاجراءات الخاصة بالدليل

DIRECTRY RELATED PROCEDURES

يوجد في نظام نبراس أربعة اجراءات قياسية خاصة في التحكم بالدليل، وهي:

١) غير_دليل

٢) اعمل_دليل

٣) احذف_دليل

٤) اسم_دليل

غير_دليل

تركيب الجملة : غير_دليل (دليل)؛

عند استعمال الاجراء مع اعطاء الدليل المطلوب (دليل).. يغير الدليل الحالي إلى آخر جديد، ويكتب اسم الدليل (دليل) على هيئة سلسلة رموز، ويمكن الانتقال إلى دليل موجود في محرك اسطوانة آخر، بحيث يكتب اسم محرك الاسطوانة مع اسم الدليل.

مثال:

غير_دليل ('ARABWORD \')؛

المثال السابق يغير الدليل الحالي إلى دليل (ARABWORD) في نفس الاسطوانة. أما اذا كان موجودا في محرك اسطوانة أخرى مثلا (:B).. يتغير المثال السابق ليصبح:

غير_دليل ('B: \ ARABWORD')

اعمل_دليل

تركيب الجملة : اعمل_دليل (دليل)؛

عند استعمال الاجراء مع اعطاء الدليل المطلوب (دليل)، يعمل دليل ثانوي أو فرعي جديد في الدليل الحالي للاسطوانة الحالية، ويكتب اسم الدليل (دليل) على هيئة سلسلة رموز كما في الاجراء السابق.

مثال:

اعمل_دليل ('ARABWORD \ ')؛

المثال السابق يعمل دليل جديد اسمه (ARABWORD) في الدليل الحالي الموجود في الاسطوانة الحالية. فإذا كانت الاسطوانة الحالية (:C) والدليل الحالي (TEST) فإن المسار لهذا الدليل من الدليل الجذري يكون (C:\TEST\ARABWORD) وإذا أردنا أن نعمل دليلا جديدا في دليل يختلف عن

الدليل الحالي، أو محرك اسطوانة آخر غير الحالية، فيجب تحديد المسار الذي يدل موقع الدليل الثانوي فيكتب اسم محرك الاسطوانة مع الدليل في المعطيات ليصبح المثال السابق:

اعمل_دليل ('B:\ARABIC\ARABWORD')؛

احذف_دليل

تركيب الجملة : احذف_دليل (دليل)؛

الاجراء يحذف الدليل المعطى بالسلسلة الرمزية (دليل) بشرط أن يكون الدليل خاليا من الملفات ويكون المستخدم في الدليل الأب لهذا الدليل، فإذا كان في دليل آخر أو اسطوانة أخرى فعلية تحديد اسطوانة ومسار الدليل.

مثال:

احذف_دليل ('\ARABIC')؛

المثال السابق يحذف الدليل (ARABIC)، فإذا كان موجودا في محرك اسطوانة يختلف عن الاسطوانة الحالية مثلا: (:B) فيجب تغيير المثال السابق ليصبح: احذف_دليل ('B:\ARABIC')

اسم_دليل

تركيب الجملة : اسم_دليل (محرك، دليل)؛

الاجراء يعطي اسم الدليل الحالي (دليل) الموجود على محرك الاسطوانة (محرك)، ويجب تعريف المتغير (دليل) بأنه سلسلة رموز والمتغير (محرك) يمثل عدد صحيح، حيث:

(٠) = محرك الاسطوانة الحالية.

(١) = A

(٢) = B

(٣) = C

(٤) = D

```
برنامج الدليل؛
    متغير
        دليل : سلسلة [٣٠]؛
        محرك : صحيح؛
بداية
    اعمل_دليل ('ARABIC\')؛
    اسم_دليل (١، دليل)؛
    اكتب_سطر (' الدليل الحالي : '، دليل)؛
    احذف_دليل ('ARABIC\')؛
نهاية.
```

١٧ــ٢ تنفيذ البرامج

PROGRAM EXECUTION

هناك اجراء خاص لتنفيذ وتشغيل برنامج من خلال برنامج آخر وهو «**نفذ**» ويجب أن يكون البرنامج المراد تنفيذه مترجما من قبل بنظام نبراس .. أي يكون برنامجا أمريا بنهاية تصنيفية من نوع (COM.)، وهذا النوع من التنفيذ يختلف عن طريقة تنفيذ الملفات الاحتوائية ــ والتي سوف نوضحها في الجزء **١٧ــ٣** فالاجراء «نفذ» يضع البرنامج المراد تنفيذه في ذاكرة الجهاز ــ مزيحاً البرنامج الحالي الذي يستخدم هذا الاجراء من الذاكرة، وفي هذه الحالة لا يمكن الرجوع إلى البرنامج الحالي، فكل برنامج يعتبر مستقلا عن الآخر.

نفذ

تركيب الجملة : نفذ (اسم_البرنامج)؛

ينفذ الاجراء البرنامج المترجم (اسم_البرنامج)، حيث يعرف كملف غير مصنف، ويجب أن يكون مخصصا من قبل لأي متغير من نوع (ملف) باستعمال الاجراء «خصص» ويكون الملف المخصص مترجما وأمريا حيث يكون له النهاية التصنيفية (COM.).

برنامج نفذ ملف؛

متغير

اسم البرنامج : **ملف** ؛

بداية

خصص (اسم البرنامج)؛ 'TEST2. COM')؛

نفذ (اسم البرنامج) ؛

نهاية.

فإذا كان البرنامج **TEST2** يحتوي على الجمل التالية :

برنامج TEST2؛

بداية

اكتب_سطر (' تجربة التنفيذ !!!!!')؛

نهاية.

فعند تنفيذ البرنامج (نفذ_ملف)... وليكن اسمه (TEST1. COM) تظهر على الشاشة الجملة «تجربة التنفيذ !!!!»، فيجب أن يكون البرنامج (TEST2. COM) موجودا في الدليل الحالي أو الاسطوانة الحالية، ويمكن كتابة اسم الاسطوانة ومسار الدليل كجزء من اسم البرنامج، واذا كان البرنامج غير موجود في الدليل أو الاسطوانة، فسوف يحصل خطأ في التنفيذ. واذا أراد المبرمج العودة إلى البرنامج السابق فيجب أن يحتوي البرنامج (TEST2. COM) قبل الخروج منه والعودة إلى البرنامج (TEST1. COM) على الجمل التالية:

خصص ('TEST1. COM' ، اسم_البرنامج)؛

نفذ (اسم_البرنامج)؛

ليصبح البرنامج TEST2 كالتالي:

برنامج TEST2؛

متغير

اسم_البرنامج : **ملف**؛

بداية

اكتب_سطر (' تجربة التنفيذ !!!!!')؛

خصص (اسم_البرنامج، 'TEST1. COM')؛

نفذ (اسم_البرنامج)؛

نهاية.

وهذا يؤدي إلى إيجاد تكرار لا نهائي بين البرنامجين حيث كل برنامج ينفذ الآخر إلى ما لا نهاية إلا إذا وجد قائمة اختيارية في أحد البرنامجين لتسمح للخروج من هذا التنفيذ، ويجب مراعاة أن عملية تنفيذ البرامج التي نهايتها التصنيفية (COM) لا تتم في نظام «نبراس».. أي عملية التنفيذ تتم عن طريق نظام التشغيل الرئيسي (DOS) وليس عن طريق النظام العربي.

١٧ــ٣ الملفات

FILES

سبق وأن تكلمنا في باب سابق عن الملفات وعلاقتها بالبرنامج، وفي هذا الموضوع سوف نتكلم عن الملفات وعلاقتها بنظام التشغيل الرئيسي. فبعض البرامج يجب أن تعمل مع ملفات بيانات خارجية،.. أي مخزنة داخل اسطوانة تخزين. والسبب الرئيسي في ذلك هو الحد الأعلى لسعة البرنامج وهو ٦٤ ألف وحدة (بايت). ونظام نبراس العربي يسمح بأستعمال السعة نفسها

للبيانات. فعندما يحتاج البرنامج والبيانات إلى أكثر من ٦٤ ألف وحدة من الذاكرة فإنه من الأفضل استعمال ملف بيانات خارجي.

البرامج التي تستعمل ملفات خاصة بالموظفين أو قائمة بالعناوين والتي تتطلب حجم ملفات كبيرة ،يجب أن تستعمل ملفات خارجية للتخزين وتأتي أهمية الملفات ليس فقط بالحجم الكبير ولكن إمكانية نقل الملف ووجوده في الاسطوانة حتى ولو كان الجهاز غير شغال، والملفات الخارجية ليست كالبرامج أو ملفات النصوص ــ إذ إنها تحتوي على مجموعة من البيانات مخصصة ومصممة فقط للبرنامج نفسه فإذا كان الملف يحتوي على عشرين ألف سجل من سجلات الموظفين فإنه لا يمكن استعمال هذه السجلات الا لبرامج تكون معرفة فيها، ولا يمكن استعمال الملفات لأي لغة أخرى إلا إذا حُوِّل الملف ليتوافق مع اللغة الأخرى. وعلى الرغم من وجوب تعريف الملفات بالبرنامج ومحتوياتها وطول كل حقل وجزء في فقرات وسجلات الملف إلا أنه لا ينبغي تحديد طوله إذ أن الملف غير مقيد بسعة ثابتة ولكنه مقيد بسعة اسطوانة التخزين فإذا كان هناك اسطوانة تخزين «مرنة».. والتي تتسع إلى ٣٦٠ ألف وحدة فإن الملف يصبح مقيد بهذه السعة، واذا كان هناك اسطوانة «صلبة» والتي تتسع إلى ١٠ ملايين وحدة أو أكثر فإن الملف يصبح مقيد بهذه السعة نفسها، وليس هناك علاقة بسعة الذاكرة الرئيسية وسعة ملفات البيانات الخارجية وهذا النوع من الملفات يسمى ثنائي **(BINARY)**. وتختلف عن الملفات المكتوبة بطريقة الشفرة القياسية **(ASCII)**. ومن الطبيعي جدا أن يعطى الملف اسم يصف محتوياته.

اسم الملف

اسم الملف في نظام التشغيل يحتوي على اسم الاسطوانة، ثم الدليل

الموجود فيه الملف واسم الملف ويفصل بينها الفاصل «\» كالآتي:

الاسطوانة، دليل، دليل...دليل، اسم الملف

إذا بدأ اسم الملف بالفاصل «\» فإن نظام الشتغيل سوف يبحث عن الملف بالدليل الحالي، وفي حالة كتابة اسم الملف بدون اسم الاسطوانة واسم الدليل فإن نظام التشغيل يفترض أن الملف موجود في الاسطوانة الحالية والدليل الحالي.

يتكون اسم الملف من رمز إلى ثمانية رموز، يليه النقطة الاختيارية (.) يتبعها ثلاثة رموز اختيارية توضح نوعية الملف فمثلا:

FILENAME. TXT

حيث (FILENAME) هو اسم الملف (TXT) النهاية التصنيفية أو تحويلة الملف. تحويلة الملف.

فبعد تأسيس الملفات الخارجية بواسطة نظام نبراس، فإن نظام التشغيل الرئيسي (DOS) يتولى عملية ادارة الملفات من حفظ ونقل والغاء وتصوير وجميع أوامر نظام التشغيل الرئيسي المتعلقة بها. والملفات الثنائية تنقسم إلى عدة أنواع فمنها ما هو قابل للتنفيذ.. أي ينفذ بواسطة نظام التشغيل الرئيسي ويكون منتهيا بالتصنيف «EXE» أو «COM»، ونظام نبراس يستطيع عمل النوع الأخير من هذه الملفات، وهناك اختلاف بين هذه الملفات ولكن لن نتطرق إليها لأنها ليست ذات أهمية في لغة نبراس، ومنها ما يكون ملف بيانات، وفي هذه الحالة تكون النهاية التصنيفية للملف إما ثابتة من قبل البرنامج كما في قاعدة البيانات «DATA BASE» فهي ثابتة على «(.DBF)» وكذلك النهاية التصنيفية (IKW) للبرنامج الجدولي «لوتس ٢٠٢١»، وكذلك ملفات تركيب وحدات النظام (CONFIGURATION. SYS)، أما نظام نبراس فيعطي المبرمج حرية

اختيار الاسم والنهاية التصنيفية لملفات البيانات ويعطي النظام للملف الأمري القابل للتنفيذ التحويلة «.COM» وجميع الملفات لا تعامل كما تعامل الملفات المشفرة أو ملفات النص.

وقد صمم بنظام اللغة، كما ذكر سابقا دوال واجراءات لتساعد المبرمج على التحكم بالملفات عن طريق استعمال نظام التشغيل (DOS) وهي:

خصص

افتح_م_ق

افتح_م_ك

ارسل_إلى_ملف

اهمل

الحق

اغلق

كما يتوفر ثلاثة برامج ثانوية إضافية خاصة بالملفات لتساعد في استعمال السجلات في الملفات وهي:

دالة **حجم_ملف_طويل**

دالة **موضع_ملف_طويل**

اجراء **بحث_طويل**

فالدالتان «حجم_ملف_طويل» و «موضع_ملف_طويل» تعملان كما تعمل الدالتان «موضع_ملف» و «حجم_ملف» ووجه الاختلاف في نوع النتيجة العائدة، حيث أن نتيجة الدالتان «موضع_ملف»، «حجم_ملف» من نوع

(صحيح)، ونتيجة الدالتان «حجم_ملف_طويل»، «موضع_ملف_طويل» من نوع (حقيقي).

وبالنسبة للاجراء «بحث طويل» فيستعمل كما يستعمل الاجراء «بحث عن» مع نفس المعطيات اللازمة إلا أن المعطى الثاني في الاجراء «بحث طويل» من نوع (حقيقي) حيث أنه في الاجراء «بحث عن» من نوع (صحيح).

بحث طويل (اسم ملف، ن)؛ (ن) من نوع حقيقي.
بحث عن (اسم ملف، ن)؛ (ن) من نوع صحيح.

وجميع هذه الاجراءات والدوال تستعمل كما في الملفات والنصوص القياسية، ومن الاجراءات التي تستعمل للتحكم بالملفات داخل الاسطوانة اجراءان يمكن استعمالهما للقيام بعمليات الحذف وتغيير الاسم وهما:

امسح / غير اسم

يجب إعداد الملف قبل استعمالهما كما لو كان مؤسس للقراءة أو الكتابة، فيخصص الملف ومن ثم يستعمل أحد الاجرائين.

وكما ذكر سابقا فإن لكل برنامج أو معالج كلمات تنسيق وتصميم خاص به ومن هذه البرامج معالج الكلمات الشهير «ورد ستار» (WORD STAR)، فهو لا يخزن الملفات حسب الشفرة (ASCII)، فإذا أريد استعمال أحد ملفاته بواسطة برنامج آخر فيجب أولا تحويله إلى ملف مشفر.

البرنامج الآتي يحول ملفات ورد ستار إلى ملفات مشفرة قياسية، مع استعمال بعض الاجراءات السابقة.

```
برنامج تحويل_ملف؛
ثابت
    اسم_مؤقت = 'عرب.عرب'؛
متغير
    اسم_ملف : سلسلة [٣٠]؛
اجراء عملية_الملف؛
متغير
    س : صحيح؛
    الملف_الداخل، الملف_الخارج : نص؛
    ع : رمز؛
بداية
    خصص (الملف_الداخل، اسم_ملف)؛
    افتح_م_ق (الملف_الداخل)؛
    خصص (الملف_الخارج، اسم_مؤقت)؛
    افتح_م_ك (الملف_الخارج)؛
    طالما عكس ن_م (الملف_الداخل) اعمل
    بداية
        اقرأ (الملف_الداخل، ع)؛
        ع := حرف (رقم (ع) باقي ١٢٨)؛
        اذا ع := حرف (١٣) اذن
            اكتب_سطر (الملف_الخارج)
        والا اذا ع>= '' اذن
            اكتب (الملف_الخارج، ع)؛
    نهاية؛                                  (* طالما *)
    اغلق (الملف_الداخل)؛
    اغلق (الملف_الخارج)؛
```

```
امسح (الملف_الداخل)؛
غير_اسم (الملف_الخارج، اسم_ملف)؛
نهاية؛ (٠ الاجراء عملية الملف ٠)
بداية
اكتب (' ادخل الملف المراد تحويله او <مفتاح العودة> للخروج؟')؛
اقرأ_سطر (اسم_ملف)؛
اذا طول (اسم_ملف)>. اذن
عملية_ملف؛
نهاية. (٠ البرنامج الرئيسي ٠)
```

ويجب اغلاق الملف قبل استخدام احد الاجرائين، وعلى المبرمج التأكد عند استعمال الاجراء (غير_اسم) أن الاسم الجديد غير موجود بالدليل المستعمل أو الاسطوانة وللتأكد، اليك هذه الدالة التي تعطي نتيجة (صح) إذا كان الملف موجودا ونتيجة (خطأ) إذا كان الملف غير موجود:

```
صنف
اسم = سلسلة [٣٠]؛
.
.
.
دالة موجود (اسم ملف : اسم) : منطقي؛
متغير م: ملف؛
بداية
خصص (م، اسم_ملف)؛
{$م+} افتح_م_ق (م)؛ {$م-}
موجود : = (نتيجة_د_خ =٠)
نهاية ؛
```

فعند استعمال الدالة، يجب ارسال «اسم ملف» إلى الدالة ويكون اسم ملف متغير خصص من قبل وتستعمل كالآتي:

اذا موجود (اسم_ملف) **اذن**
اكتب_سطر (' الملف موجود')
والا اكتب_سطر ('الملف غير موجود')؛

١٧ــ٣ــ١ الملفات الاحتوائية

INCLUDE FILES

لمعالج الكلمات الداخلي سعة معينة يمكن أن يحتويها من النص وهي ٦٤ ألف بايت، وعند وصول النص إلى هذا الحد تظهر على الشاشة جملة تحذيرية تدل على أن الملف أوشك على الوصول للحد الأعلى من السعة النظامية. فإذا كان النص أكبر من ٦٤ ألف بايت فإن الذاكرة المخصصة للمحرر لن تتسع لذلك الملف، والجملة «**الملف كبير لا يمكن التحرير**» تظهر على الشاشة. والطريقة الوحيدة للتخلص من المشكلة هي تقسيم الملف أو البرنامج إلى عدة ملفات أو برامج وتخزينها في الاسطوانة، ثم استعمال الموجه {م$} يليها اسم البرنامج المراد طلبه في البرنامج الأساسي، ويعامل كما لو كان جزء من البرنامج، ولكن المترجم يحوي فقط هذا الملف عند التنفيذ.

وللملفات الاحتوائية فوائد ليست فقط لذات الحجم الكبير، ولكن لاحتواء المترجم على أي ملف كان، فإذا كان هناك مكتبة من الدوال والاجراءات وتستعمل في أكثر من برنامج، ففي هذه الحالة لا تتطلب كتابة الاجراءات

والدوال مرة أخرى في البرنامج، ولكن تكتب فقط مرة واحدة، وعند الحاجة توضع كملفات احتوائية.

فمثلا إذا كان هناك اسم ملف «**مكتبة.عرب**» وأردنا احتوائها مع البرنامج عند التنفيذ، فيجب استعمال الموجهة {$م} مع اسم الملف {$م مكتبة .عرب} ويجب ترك فراغ بين الموجهة واسم الملف، كما يمكن استعمال (* $م **مكتبة.عرب** *) فعند وصول المترجم إلى موجهة الملفات الاحتوائية، يقرأ الملف كما لو كان جزء من البرنامج الرئيسي كما يمكن للملفات الاحتوائية نفسها أن تحتوي على ملفات احتوائية أخرى، والحد الأعلى للملفات الاحتوائية هو ٨ ملفات متداخلة، ولا يمكن وضع الملفات الاحتوائية داخل (**بداية**) و (**نهاية**). كما يجب أن يكون الملف الاحتوائي موجود في الدليل الحالي والاسطوانة الحالية، أو وضع الدليل والاسطوانة كجزء من اسم الملف {$م A:\عرب\مكتبة.عرب}، واذا كان الملف غير موجود ستظهر الجملة التحذيرية «**خطأ ١٥ الملف غير موجود**». ولا يتطلب وضع النهاية التصنيفية للملف إذا كانت (عرب.) حيث يعتبرها المترجم النهاية النظامية للملف، فإذا كان اسم الملف المخزن في الاسطوانة «برامج.عرب»، ووضع الملف الاحتوائي {$م برامج} فإن المبرمج يبحث عن (برامج.عرب).

مثال:

اذا كانت البرامج الثانوية الآتية مخزنة في الاسطوانة تحت اسم «**مكتبة. عرب**»

```
اجراء ع_س (ع، س : صحيح)؛
بداية
    اذهب_ع _س (ع، س)؛
نهاية؛
```

```
اجراء لون (لون_امامي، لون_خلفي : صحيح)؛
بداية
    لون_نص (لون_امامي)؛
    خلف_نص (لون_خلفي)؛
نهاية؛
اجراء كتابة (عمود، سطر، لون_امامي، لون_خلفي، صحيح؛ ن : سلسلة)؛
بداية
    ع_س (عمود، سطر)؛
    لون (لون_امامي، لون_خلفي)؛
    اكتب_سطر ( ن )؛
نهاية؛
```

فإذا أردنا استعمال هذه البرامج في برنامج آخر فيوضع ملفها كملف احتوائي داخل البرنامج كالآتي:

```
برنامج تجربة_الملف_الاحتوائي؛
    متغير
        اسم : سلسلة؛

{$م مكتبة}
بداية
    ع_س (١٠، ٢٠)؛ (٭ نلاحظ ان هذا الاجراء موجود في الملف الاحتوائي ٭)
    اكتب_سطر ('الرجاء ادخال الاسم')؛
    اقرأ_سطر (اسم)؛
    كتابة (٥، ٤، ١، ٤، اسم)؛ (٭ وأيضا هذا الاجراء ٭)
نهاية.
```

كما يمكن استعمال أكثر من ملف احتوائي في برنامج واحد مثل:

```
برنامج تجربة الملف الاحتوائي؛
متغير
    اسم : سلسلة؛
{$م برنامج١. عرب}
{$م برنامج٢. ملف}
{$م PROGRAM. PAS}
بداية
.........
.........
نهاية.
```

١٧ــ٤ البرامج الثانوية الخارجية

EXTERNAL SUBPROGRAMS

البرامج الثانوية الخارجية عبارة عن اجراءات أو دوال كتبت من قبل سواء بهذه اللغة أو بلغة أخرى، ومترجمة إلى لغة الحاسب الآلي «**MACHINE LANGUAGE**». ومن الأفضل استعمال اللغة الانجليزية عند كتابة اسم البرنامج. فقد صمم المترجم للاستفادة من هذه البرامج لزيادة قدرة وامكانية هذه اللغة، وبالإمكان كتابة أي برنامج كان باللغة العربية مع استعمال البرامج الخارجية. ولكن قبل استعمال أي برنامج خارجي، لا بد من معرفة الطريقة السليمة لاستعمال هذه البرامج.

فاستعمالها يتطلب معرفة تامة وعالية بالحاسب الآلي. فترجمة أي برنامج كان إلى لغة الحاسب الآلي تمكن من استخدامه مع أي برنامج كتب بهذه اللغة، وباستعمال كلمة «**خارجي**»، يدرك المترجم أن هذا البرنامج مترجم من قبل وموجود في مكان آخر فعند استعمال هذه الكلمة يمكن تنفيذ برنامج بلغة الحاسب الآلي، ويمكن طلب البرنامج بواسطة برنامج آخر مكتوب بلغة

نبراس، فعند ترجمة نبراس للبرنامج فإنه يأخذ البرنامج المترجم ويضعه مع برنامجه الأساسي كجزء منه. وكلمة «**خارجي**» مهمة خاصة عند طلب ملف مترجم لملف مستقل يعمل مع الحاسب الآلي ويحتاج إلى اللغة العربية، ولذلك يمكن طلب البرنامج الذي يحمل الحروف العربية ويضعها في ذاكرة الحاسب الآلي حتى يتسنى تنفيذ البرنامج بمفرده وعدم اللجوء إلى نبراس لتشغيله.

مثال:

يتوفر في اسطوانة نبراس برنامج مترجم إلى لغة الحاسب الآلي وهو «**حروف.ث**» ومهمته وضع الحروف العربية في ذاكرة الحاسب الآلي، فعند كتابة أي برنامج يجب استعمال هذا الاجراء الداخلي لتسهيل عملية وضع الحروف العربية في ذاكرة الحاسب الآلي.

```
    برنامج تجربة الخارجي
      اجراء عربي : خارجي 'حروف.ث'؛
بداية                                    (ه بداية البرنامج وليس الاجراء ه)
    عربي؛                                (ه لشحن الحروف العربية إلى الذاكرة ه)
نهاية.                                   (ه نهاية البرنامج ه)
```

١٧ــ٥ موجهات المترجم

COMPILER DIRECTIVS

من أهم عوامل البرمجة هي قدرة المبرمج من بحث وتصحيح الأخطاء الموجودة في البرنامج. فإذا كانت الأخطاء ناتجة من تركيب الجملة فهي بسيطة، إذا أن المترجم أثناء قيامه بعملية التنفيذ يقوم بكتابة الخطأ المتوقع مع رقم السطر الذي وقع عليه الخطأ، أما إذا كان الخطأ منطقي، وعادة ما يكون من أصعب الأخطاء، لأنه من الصعب تحديد الجملة المنطقية التي وقع فيها

الخطأ، ولا يقوم المترجم باصدار جملة تحذيرية ويستمر في التنفيذ، ولكن المبرمج يلاحظ أن المعلومات الخارجة ليست المطلوبة، لأنها ناتجة من تركيب خاطيء في المنطق مع ان تركيب الجملة صحيح. فنجد البرنامج ينفذ بدون خطأ فمثلا إذا أدخل المبرمج سلسلة رمزية أطول من السلسلة المعرفة في البرنامج ووضعها في منظومة ثم أراد البحث عنها في المنظومة فإن اجراء البحث لن يجد السلسلة ونجد أن البرنامج ينفذ بدون أخطاء في تركيبه ولكن لن يتمكن البرنامج من البحث، ونتج عن ذلك عملية اخراج خاطئة، ولهذا السبب بني في المترجم عدة موجهات (DIRECTIVES) لتساعد المبرمج للتعرف على الأخطاء وأنواعها.

وبما أن استعمال الموجهات يسبب تأخير بسيط في عملية التنفيذ.. حيث تزيد من وقت التنفيذ المخصص وذلك بفحص صحة الجمل التنفيذية، لهذا أختيرت الحالة النظامية للموجهات عند التنفيذ لتكون الحالة الغير فعالة، ويمكن تشغيل الموجهة بجعلها موجبة (+) في منطقة معينة والغاؤها في منطقة أخرى، ولكن هناك بعض الموجهات لا تستعمل إلا مرة واحدة فقط في البرنامج، فتستعمل إما موجبة (+) أو سالبة (−) وتستمر هذه الحالة أثناء عملية التنفيذ، وهذه الموجهات هي (ؤ، ع) ويجب وضعها بين الأقواس المائلة فتبدأ بكتابة القوس الأيمن «{» ثم اشارة «$» ثم رمز الموجهة الذي يتكون من أحد الحروف (ع، م، ر، ؤ)، ثم يكتب القوس الأيسر «}».

وطريقة استعمال الموجهات وأغراضها كالآتي:

أ ــ الموجهة { م$ }:

تستعمل الموجهة لفحص جميع عمليات الادخال والاخراج، فإذا حصل خطأ منطقي في عملية ادخال أو اخراج، فإن البرنامج المترجم يخرج مع اشعار

المستعمل بأن خطأ قد حدث في عملية (د/خ)، أما إذا كان البرنامج مترجم إلى برنامج أمري «COM.» فإن المترجم سوف يرجع إلى نظام التشغيل الرئيسي مع اشعار المستعمل بهذا الخطأ، وللتخلص من المشكلة وجعل البرنامج ينبه المستعمل بحدوث خطأ في عملية (د/خ)، واستمرار التنفيذ باستعمال الدالة (نتيجة_د_خ) مع الموجهة {$م+}، وتلغى هذه الموجهة بعد عملية الفحص باستعمال الموجهة السالبة {$م-}.

```
برنامج فحص_عملية_د/خ؛
    صنف
        سلسلة_رموز = سلسلة [٦٠]؛
    متغير
        اسم_الملف : سلسلة_رموز؛
دالة فحص_ملف (اسم : سلسلة_رموز): منطقي؛
    متغير م : ملف؛
    بداية
        خصص (م، اسم)؛
        {$م+} افتح_م_ق (م)؛ {$م-}
        فحص_ملف := (نتيجة_دخ=٠) ؛
    نهاية؛
بداية (* البرنامج الرئيسي *)
    كرر
        اكتب (' اسم الملف المراد فحصه < مفتاح الادخال للخروج> :')؛
        اقرأ (اسم_الملف)؛
        اذا طول (اسم_الملف) >٠ اذن
            اذا فحص_ملف (اسم_الملف) اذن
                اكتب_سطر (' الملف الموجود')
```

والا اكتب_سطر ('الملف غير موجود')؛
حتى طول (اسم_الملف) = ٠
نهاية.

البرنامج السابق يوضح استعمال الموجهة {$م} لفحص عملية فتح ملف ويستخدم الدالة «نتيجة د خ» التي تعطي صفرا إذا كانت عملية الادخال أو الاخراج صحيحة، فإذا تمت عملية فتح الملف فإن نتيجة الدالة «فحص ملف» ستصبح صحيحة ويكون الملف فالملف موجود على اسطوانة التخزين. كما يمكن استعمال الموجهة {$م} بدون إشارات لاحتواء برنامج آخر عند التنفيذ... وهذه الموجهة ترشد المبرمج لفتح الملف المطلوب احتوائه وضمه مع البرنامج المنفذ فقط في فترة التنفيذ، وبعد الانتهاء من التنفيذ فإن جميع محتويات الملف الإحتوائي تكون جزء من البرنامج الرئيسي.

والملفات الإحتوائية عادة ما تكون مجموعة من الاجراءات والدوال شبه قياسية وتستعمل بكثرة في أكثر من برنامج، وتكون شبه قياسية لأنها لا تخضع إلى برنامج معين بل تستعمل مع جميع البرامج بدون الحاجة إلى تغييرها (راجع الباب المخصص بالبرامج الاحتوائية). ويجب أن تكون الدوال خالية من أي خطأ، وأفضل طريقة لاختبار الملفات الاحتوائية هي : معاملتها كما لو كانت برامج كاملة فإذا نفذ البرنامج بدون أخطاء تصمم على هيئة وتركيب البرنامج الاحتوائي.

ب ــ الموجهة {$ؤ}

تستعمل الموجهة للتحكم في لوحة المفاتيح أثناء وبعد عملية التنفيذ، والحالة النظامية لهذه الموجهة هي الحالة الموجبة {$ؤ+} للسماح للمبرمج بالتحكم بلوحة المفاتيح، وتتحكم الموجهة بايقاف عملية اظهار الرموز على الشاشة في حالة الاخراج بالضغط على المفتاحين <Ctrl>

<s> ، وهذا الاختيار مهم جدا إذا كانت البيانات الخارجة أكثر من ٢٥ سطرا حيث يمكن إيقاف تدرج المعلومات والنظر إليها، فالحاسب الآلي سريع مما يجعل المستخدم لا يستطيع رؤية وقراءة جميع البيانات الخارجة، فعملية الإيقاف تمكن المستخدم من التأكد من صحة البيانات ثم الاستمرار في عملية اخراج المعلومات بضغط نفس المفتاحين مرة أخرى. ويمكن كذلك استعمال المفتاحين <Ctrl> <c> لقطع أي عملية تنفيذ أو اخراج معلومات، وهذه العملية مهمة كذلك إذ أنها تتيح للمبرمج إمكانية قطع البرنامج والتوقف عن أي عملية كان البرنامج ينفذها وخاصة عندما يكون البرنامج قد وقع في عملية التكرار اللانهائي والتي يقع فيها الكثير من المبرمجين المبتدئين، أما عند استعمال الموجهة السالبة {ؤ-$} فإن العملية ستكون لاغية المفعول.

ج ــ الموجهة {ع$} :

للتحكم في لوحة المفاتيح عند التنفيذ، وهذه الموجهة لا تؤثر على عملية إيقاف أو استمرار اظهار البيانات، ولكن لها علاقة فعالة بمقاطعة البرنامج والحالة النظامية لهذه الموجهة هي الحالة الموجبة {ع+$} ، والتي لا تسمح لعملية مقاطعة البرنامج بينما الحالة النظامية للموجهة {ؤ$} تسمح لهذه العملية. ويمكن إختصار العملية وإزالة الإلتباس.. بمعرفة انه عندما تكون الموجهتان مختلفتان.. أي احداهما موجبة (+) والأخرى سالبة (−) فإن عملية إيقاف تنفيذ البرنامج فعالة مع إهمال الحالة النظامية لهاتين الموجهتين.. وعند استعمال الموجهتين مع بعضهما نحصل على ثلاثة حالات وهي:

١) {ع-$} ، {ؤ+$} وهي الحالة النظامية، وتسمح لعملية ايقاف الرموز على الشاشة <Crtl> <s> وعملية المقاطعة <Crtl> <c>

٢) {$ع-} ، {$ؤ-} العمليتان لاغيتان.

٣) {$ع+} ، {$ؤ-} عملية إيقاف الرموز غير فعالة <Ctrl><S> ، ولكن عملية مقاطعة التنفيذ فعالة <Ctrl><C> .

الموجهتان {$ع} ، {$ؤ} غير قابلتين للتغيير.. فعند وضع إحداهما سالب والأخرى موجب لا يمكن تغييرهما في مكان آخر في البرنامج.

د ـ الموجهة {$ر} :

وتفيد هذه الموجهة عند استعمال السلاسل الرمزية، فهي تفحص طول السلسلة المرسلة إلى الاجراء عن طريق المعطيات. فعندما تكون هذه الموجهة بالحالة الموجبة {$ر+} (فعالة)، فإن المترجم سيقوم بعملية فحص السلسلة المرسلة إلى الاجراء عن طريق متغير المعطيات، وللتأكد بأن طول السلسلة يتوافق مع طول سلسلة الاجراء المطلوب، وإذا كانت الموجهة في الحالة السالبة {$ر-} ، فإن عملية الفحص لن تتم ويسمح بارسال السلاسل متعددة الطول إلى الاجراء، وستكون النتائج غير متوقعة إذا كانت السلسلة طويلة جدا على الاجراء المقصود.

هـ ـ الموجهة {$ق} :

تستعمل هذه الموجهة للتحكم في فحص عملية الفهرسة في البيانات الهيكلية كالمنظومات والمؤشرات. عندما تكون الموجهة الموجبة {$ق+} فعالة، فإن جميع العمليات الخاصة بفهارس المنظومات تفحص للتأكد ان الفهارس واقعة بين المدى الأدنى والأعلى، وكذلك جميع عمليات التخصيص إلى المقاييس. وعندما تكون الموجهة سالبة {$ق-} ، فإن

عملية الفحص لن تتم، وفي هذه الحالة سيحدث خطأ منطقي في الفهرسة مما يسبب للبرنامج أخطاء غير متوقعة، والأفضل جعل الموجهة فعالة عند القيام بعملية البرمجة وفحص الأخطاء، وعند الانتهاء تعاد إلى الحالة الغير فعالة.

ملحق « أ »

برمجة المفاتيح الوظيفية PROGRAMMING THE FUNCTION KEYS

عند الضغط على أحد مفاتيح اللوحة، فإن المفتاح ينتج شفرة معينة يرسلها عن طريق لوحة المفاتيح إلى وحدة المعالجة المركزية (CPU) وبدوره يترجمها ويرسل الرمز المقابل لها إلى الشاشة. ولكن يوجد بعض الرموز التي تعطي أكثر من شفرة واحدة عند ضغطها مع مفتاح آخر في آن واحد، فمثلا عند ضغط المفتاح «ل» فإن الشفرة تكون ٢٥١ وعند ضغط المفتاح «ل» مع مفتاح (Shift) فإن رقم الشفرة يتغير إلى ١٥٣، وعند استعمال المفتاحان «Ctrl» و «Alt» مع احد الرموز فانها تنتج شفرات مختلفة وهناك بعض المفاتيح كالمفاتيح الوظيفية ومفاتيح الأرقام الموجودة على الجهة اليمنى من لوحة المفاتيح تنتج شفرتان شفرة المسح (SCAN CODE) وهي الرقم (#٢٧) وشفرة أخرى للدلالة على أن هناك مفتاح قد ضغط.

ولقراءة هذه المفاتيح يستعمل الاجراء كما يلي:

```
اجراء   قراءة_مفاتيح (متغير مفتاح : رمز ؛ متغير مفتاح_مضغوط : منطقي)؛
بداية
        مفتاح_مضغوط : = خطأ؛
        اقرأ (ل_م، مفتاح)؛
        اذا (مفتاح = #٢٧) و ضغط_مفتاح اذن
        بداية
            اقرأ (ل_م، مفتاح)؛
            مفتاح_مضغوط : = صح؛
        نهاية؛
نهاية؛
```

عندما تكون القيمة المنطقية للمتغير (مفتاح مضغوط) تساوي «صح»، فإن مفتاح ذو شفرتين قد ضغط، وعندما تكون القيمة المنطقية للمتغير (مفتاح مضغوط) تساوي «خطأ» ، فإن مفتاح ذو شفرة واحدة فقط قد ضغط.

البرنامج الآتي يعطي الرمز ورقم الشفرة لأي مفتاح.. سواء كان بشفرتان أو بشفرة، أو الضغط على مفتاحين في آن واحد.

```
برنامج لوحة_المفاتيح؛
  متغير
      مفتاح          :رمز؛
      مفتاح_مضغوط :منطقي؛
بداية
  اكتب (' اضغط على أي مفتاح أو <Esc> للخروج')؛
  كرر
      مفتاح_مضغوط : = خطأ؛
      اقرأ (ل_م، مفتاح)؛
      اذا (مفتاح = # ٢٧) و (ضغط_مفتاح) اذن
      بداية
          مفتاح_مضغوط := صح؛
          اقرأ (ل_م، مفتاح)؛
      نهاية؛
      اكتب (' مفتاح مضغوط = '،مفتاح_مضغوط، 'الحرف = '، مفتاح)؛
      اكتب سطر (' رقم الشفرة = '،رقم (مفتاح))؛
  حتى (عكس مفتاح_مضغوط) و (مفتاح = #٢٧)؛ {* مفتاح Esc قد ضغط *}
نهاية.
```

		مضغوط مع		
المفتاح	مضغوط	SHIFT	Ctrl	Alt
F1	٢٧ ٥٩	٢٧ ٨٤	٢٧ ٩٤	٢٧ ١٠٤
F2	٢٧ ٦٠	٢٧ ٨٥	٢٧ ٩٥	٢٧ ١٠٥
F3	٢٧ ٦١	٢٧ ٨٦	٢٧ ٩٦	٢٧ ١٠٦
F4	٢٧ ٦٢	٢٧ ٨٧	٢٧ ٩٧	٢٧ ١٠٧
F5	٢٧ ٦٣	٢٧ ٨٨	٢٧ ٩٨	٢٧ ١٠٨
F6	٢٧ ٦٤	٢٧ ٨٩	٢٧ ٩٩	٢٧ ١٠٩
F7	٢٧ ٦٥	٢٧ ٩٠	٢٧ ١٠٠	٢٧ ١١٠
F8	٢٧ ٦٦	٢٧ ٩١	٢٧ ١٠١	٢٧ ١١١
F9	٢٧ ٦٧	٢٧ ٩٢	٢٧ ١٠٢	٢٧ ١١٢
F10	٢٧ ٦٨	٢٧ ٩٣	٧٢ ١٠٣	٧٢ ١١٣
سهم يسار	٢٧ ٧٥	٥٢	٢٧ ١١٥	٢٧ ١٧٨
سهم يمين	٢٧ ٧٧	٥٤	٢٧ ١١٦	٢٧ ١٨٠
سهم أعلى	٢٧ ٧٢	٥٦	٢٧ ١٦٠	٢٧ ١٧٥
سهم اسفل	٢٧ ٨٠	٥٠	٢٧ ١٦٤	٢٧ ١٨٣
HOME	٢٧ ٧١	٥٥		٢٧ ١٧٤
End	٢٧ ٧٩	٤٩	٢٧ ١١٧	٢٧ ١٨٢
Pgup	٢٧ ٧٣	٥٧	٢٧ ١٣٢	٢٧ ١٧٦
Pgdn	٢٧ ٨١	٥١	٢٧ ١١٨	٢٧ ١٨٤
Ins	٢٧ ٨٢	٤٨	٤٧ ١٦٥	٢٧ ١٨٥
Del	٢٧ ٨٣	٤٦	٢٧ ١٦٦	٢٧ ١٨٦
Esc	٢٧	٢٧	٢٧	
Back Sp	٨	٨	١٢٧	
Tab	٩	٢٧ ١٥		
Enter	١٣	١٣	١٠	

جدول أ ــ ١ شفرة المفاتيح الوظيفية

ملحق «ب»

الأخطاء الناتجة أثناء عملية التنفيذ
COMPILER ERROR MESSAGES

---

١ ';' متوقعة

٢ ':' متوقعة

٣ '،' متوقعة

٤ '(' متوقعة

٥ ')' متوقعة

٦ '=' متوقعة

٧ ':=' متوقعة

٨ '[' متوقعة

٩ ']' متوقعة

١٠ '.' متوقعة

١١ '..' متوقعة

١٢ بداية متوقعة

١٣ اعمل متوقعة

١٤ نهاية متوقعة

١٥ من متوقعة

١٦ اجراء أو دالة متوقعة

١٧ اذن متوقعة

١٨ إلى أو تنازلي متوقعة

٢٠ عبارة منطقية متوقعة

٢١ متغير ملف متوقع

٢٢ متوقع عدد صحيح ثابت

٢٣ متوقع عبارة عدد صحيح

٢٤ متوقع متغير عدد صحيح

٢٥ متوقع عدد صحيح أو عدد حقيقي ثابت

٢٦ متوقع عبارة عدد صحيح أو عدد حقيقي

٢٧ متوقع متغير عدد صحيح أو عدد حقيقي

٢٨ متوقع متغير مؤشر

٢٩ متوقع متغير سجل

٣٠ متوقع صنف بسيط

٣١ متوقع عبارة بسيط

٣٢ متوقع ثابت سلسلة

٣٣ متوقع عبارة سلسلة

٣٤ متوقع متغير سلسلة

٣٥ متوقع ملف نص

٣٦ متوقع صنف معرف

٣٧ متوقع ملف غير مصنف

٣٨ عنوان غير معرف

محاولة الذهاب إلى عنوان غير معرف من قبل.

٣٩ الرمز غير معرف أو خطأ في الكتابة

الرمز غير معرف سواء كان عنوان، ثابت صنف، متغير حقل، أو خطأ في كتابة الرمز.

٤٠ مؤشر غير معرف في تعريف الصنف

تعريف صنف المؤشر السابق يحتوي على اسناد إلى صنف غير معروف

٤١ اسم تعريف أو عنوان مكرر

العنوان أو اسم التعريف سبق وان استعمل من قبل.

٤٢ الصنف غير مطابق

١) المتغير غير مطابق للصنف المخصص له.

٢) المعطيات الفعلية والأساسية المرسلة إلى برنامج فرعي غير متطابقة.

٣) الصنف المعرف غير مطابق لصنف الفهرس في تخصيص المنظومة.

٤) صنف العوامل في التخصيص غير مطابقة.

٤٣ ثابت خارج المدى

٤٤ الثابت ومتغير الاختيار في عبارة «حالة» غير متطابقة.

٤٥ صنف العوامل غير مطابق للعامل

٤٦ نتيجة الصنف غير صحيحة

الصنف الصحيح : المقياس، سلسلة، مؤشر

٤٧ طول السلسلة غير صحيح

طول السلسلة يجب أن يكون في المدى **١..٢٥٥.**

٤٨ طول سلسلة الثابت لا تتطابق مع الصنف.

٤٩ صنف الأساس في المدى الفرعي غير صحيح

صنف الأساس الصحيح هو جميع أصناف المقاييس ما عدا الحقيقي.

٥٠ الحد الأدنى > من الحد الأعلى

قيمة الحد الأعلى يجب ان يكون مساوي أو أكبر من قيمة الحد الأدنى.

٥١ كلمة محجوزة

لا يمكن استعمال الكلمة المحجوزة كاسم للتعريف.

٥٢ تخصيص غير قانوني

٥٣ طول السلسلة الثابتة مجاوز للسطر

طول السلسلة يجب أن لا يتعدى السطر

٥٤ خطأ في العدد الصحيح الثابت

العدد الصحيح الثابت ليس في المدى المحدد من قبل البرنامج وهو العدد ـ ٣٢٧٦٨.. ٣٢٧٦٨.

٥٥ خطأ في العدد الحقيقي الثابت

٥٦ رمز غير قانوني في اسم التعريف

٥٧ استعمال الثابت غير مسموح

٥٨ استعمال الملفات والمؤشرات غير مسموح

٥٩ المتغيرات الهيكلية غير مسموحة هنا

٦٠ استعمال ملفات النص غير مسموح هنا

٦١ استعمال ملفات النص والملفات غير المصنفة غير مسموح

٦٢ الملفات الغير مصنفة غير مسموح بها هنا

٦٣ عملية د/خ غير مسموح بها

المتغيرات من هذا النوع غير قابل للادخال والاخراج

٦٤ الملفات يجب أن تكون معطيات متغير

٦٥ فقرات الملف يجب أن لا تكون من صنف ملف

٦٦ خطأ في تركيب الحقول

٦٧ صنف اساس المجموعة خارج المدى

اساس المجموعة يجب ان يكون من نوع مقياس بحيث يكون اقل من

٢٥٥ قيمة، أو مدى فرعي يكون في حدود ... ٢٥٥

٦٨ **عبارة «اذهب_إلى» غير صحيحة**

لايمكن اسناد (**اذهب_الى**) إلى عنوان خارج عملية التكرار الحالية.

٦٩ **العنوان خارج الكتلة الحالية**

(اذهب_الى) يجب ان لا تشير إلى عنوان خارج الكتلة الحالية.

٧٠ **اجراء أمامي غير معروف**

برنامج فرعي معرف تعريف أمامي، ولكنه غير موجود.

٧١ **الملف غير موجود**

الملف الاحتوائي غير موجود

٧٢ **نهاية للبرنامج غير متوقعة**

البرنامج لا يمكن ان ينتهي بهذه الحالة، فهو يحتوي على **بداية** أكثر من **نهاية**.

٧٣ **موجهة المترجم غير صحيحة**

٧٤ **تعدد في (مع) المتداخلة**

٧٥ **نقص في الذاكرة**

يحدث هذا الخطأ عند المحاولة لحجز جزء من الذاكرة لمتغير أكثر من كمية الذاكرة المتبقية.

٧٦ **نقص في ذاكرة المترجم**

لا يوجد كمية ذاكرة كافية لترجمة البرنامج. يحدث الخطأ ولو يبدو أن هناك ذاكرة كافية، فقد تستهلكها الرزمة، والطريقة السليمة هو تجزئة البرنامج إلى عدة برامج واحتوائها في البرنامج.

OOOOOOO

ملحق «ج»

LANGUAGE SYNTAX تركيب اللغة

يوضح هذا الملحق تركيب اللغة مستخدما هيئة «باكس — نور» وقد استخدمت بعض الرموز الخاصة بهذه الهيئة وليست خاصة باللغة وهي:

::= تعني «معرفة مثل»

: تعني «أو».

{ } تعني أن المحتويات داخل الأقواس تكون مكررة أكثر من مرة.

{،} تعني أن المحتويات داخل الأقواس قد تكون مكررة أكثر من مرة ويفصل بينها الفاصلة (،).

{؛} تعني أن المحتويات داخل الأقواس قد تكون مكررة أكثر من مرة ويفصل بينها الفاصلة المنقوطة (؛).

أما الرموز الأخرى فهي تابعة للغة، والكلمات المحجوزة مكتوبة بحروف غامقة.

اسم_تعريف ::= حرف { حرف أو رقم }

اسم_تعريف_برنامج ::= اسم_تعريف

اسم_تعريف_ثابت ::= اسم_تعريف

اسم_تعريف_ثابت_مصنف ::= اسم_تعريف

اسم_تعريف_حقل ::= اسم_تعريف

اسم_تعريف_صنف ::= اسم_تعريف

اسم_تعريف_دالة ::= اسم_تعريف

اسم_تعريف_متغير ::= اسم_تعريف

اسم_تعريف_ملف ::= اسم_تعريف

اسناد_عداد_موقع ::= X: اشارة_ثابت

اشارة ::= + : −

برنامج ::= كتلة_مقدمة_برنامج

تتابع_رقم ::= رقم {رقم}

تعبير ::= تعبير_بسيط {عامل_مقارنة تعبير_بسيط}

تعبير_بسيط ::= مصطلح {مصطلح_عامل_جمع}

تعريف_اجراء ::= كتلة_مقدمة_اجراء؛

تعريف_اجراء_أو_دالة ::= تعريف_اجراء : تعريف_دالة

تعريف_ثابت ::= تعريف_ثابت_غير_مصنف : تعريف_ثابت_مصنف

تعريف_ثابت_غير_مصنف ::= اسم_تعريف = ثابت

تعريف_صنف ::= اسم_تعريف = صنف

تعريف_متغير ::= قائمة_اسم_تعريف : صنف :

قائمة_اسم_تعريف : صنف_عند_ثابت

تعريف_دالة ::= كتلة_مقدمة_دالة؛

تتابع_سادس_عشري ::= {سادس عشري}

تعريف_ثابت_مهيكل ::= اسم_تعريف : صنف = ثابت_مهيكل

ثابت ::= عدد_بدون_اشارة: اشارة عدد_بدون_اشارة: اسم_تعريف_ثابت:

اشارة اسم_تعريف_ثابت: سلسلة

ثابت_بدون_اشارة ::= عدد_بدون_اشارة: سلسلة : اسم_تعريف_ثابت :

فارغ

ثابت_مهيكل ::= ثابت : منظومة_ثابتة : سجل_ثابت: مجموعة_ثابتة

جملة_آنية ::= **آني** قائمة_عنصر_آني {، قائمة_عنصر_آني}

جملة_اجراء ::= اسم_تعريف_اجراء : اسم_تعريف_اجراء (معطيات_فعلية {، معطيات_فعلية}

جملة_كرر ::= **كرر** جملة { ؛جملة } **حتى** تعبير

جملة_تكرار ::= جملة_طالما : جملة_كرر : جملة_حيث

جملة_بسيطة ::= جملة_تخصيص : جملة_اجراء : جملة_اذهب_إلى : جملة_آنية : جملة_فارغة

جملة ::= جملة_بسيطة : جملة_مهيكلة

جملة_مهيكلة ::= جملة_مركبة : جملة_شرطية: جملة_تكرار : جملة_مع

جملة_طالما ::= **طالما** تعبير **اعمل** جملة

جملة_مع ::= **مع** قائمة_متغير_سجل **اعمل** جملة

جملة_حالة ::= **حالة** تعبير **من** عنصر_حالة { ؛ عنصر_حالة } **نهاية**: **حالة** تعبير **من** عنصر_حالة { ؛جملة حالة } **والا** جملة {؛ جملة } **نهاية**

جملة_مركبة ::= **بداية** جملة { ؛جملة } **نهاية**

جملة_شرطية ::= جملة_اذا : جملة_حالة

جملة_فارغة ::= لا شيء

جملة_حيث ::= **حيث** متغير_تحكم := قائمة_حيث **اعمل** جملة

جملة_اذهب_إلى ::= **اذهب_إلى** عنوان

جملة_اذا ::= **اذا** تعبير **اذن** جملة { **والا** جملة }

جملة_تخصيص ::= متغير::= تعبير : اسم_تعريف_دالة ::= تعبير

حرف ::= حرف_عربي : حرف_لاتيني

حرف_أو_رقم ::= حرف : رقم

حقيقي_بدون_اشارة ::= تتابع_رقم, تتابع_رقم:

تتابع_رقم, تتابع_رقم **ق** معامل_قياس:

تتابع_رقم **ق** معامل_قياس

رقم ::= ٠ : ١ : ٢ : ٣ : ٤ : ٥ : ٦ : ٧ : ٨ : ٩

رمز_تحكم ::== صحيح_بدون_اشارة : ^ رمز

سادس_عشري := رقم :A :B :C :D :E :F

سجل_ثابت ::= (عنصر_سجل_ثابت { ، عنصر_سجل_ثابت })

سلسلة ::= { عنصر سلسلة }

سلسلة_نص ::= ' { رمز }'

صحيح_بدون_اشارة ::= تتابع_رقم : $ تتابع_سادس_عشري

صنف ::= صنف_بسيط : صنف_مهيكل : صنف_مؤشر

صنف_مهيكل_غير مكثف ::= صنف_سلسلة : صنف_منظومة :

صنف_سجل: صنف_مجموعة : صنف_ملف

صنف_سلسلة ::= سلسلة_نص : رمز_تحكم

صنف_سلسلة ::= **سلسلة** { ثابت }

صنف_مهيكل ::= صنف_مهيكل_غير_مكثف : **مكثف**

صنف_مهيكل_غير_مكثف

صنف_مقياس_جزئي ::= ثابت .. ثابت

صنف_مجموعة ::= **مجموعة من** صنف_اساس

صنف_نتيجة ::= صنف_اسم_تعريف

صنف_مقياس ::= (اسم_تعريف { ، اسم تعرّيف })

صنف_مؤشر ::= صنف_اسم_تعريف

صنف_منظومة ::= **منظومة** [صنف_الفهرس { ، صنف_الفهرس}] **من**
صنف_عضو

صنف_اساس ::= صنف_بسيط

صنف_عضو ::= صنف

صنف_ملف ::= **ملف من** صنف

صنف_فهرس ::= صنف_بسيط

صنف_سجل ::= **سجل** قائمة_متغير_سجل **نهاية**

صنف_بسيط ::= صنف_مقياس : صنف_مقياس_جزئي: صنف_اسم_تعريف

عامل_جمع ::= + : − : **او** : **م_او**

عامل ضرب ::= × : ÷: **باقي تقسيم** : **و**

عامل_مقارنة ::= = : < : > : <= : >= : <> : **في**

عدد_بدون_اشارة ::= صحيح_بدون_اشارة : حقيقي_بدون_اشارة

عنصر_حالة ::= قائمة_حالة : جملة

عنصر_سجل_ثابت ::= اسم_تعريف_حقل : ثابت_مهيكل

عنصر_قائمة_حالة ::= ثابت : ثابت .. ثابت

عنصر_مجموعة_ثابتة ::= ثابت : ثابت.. ثابت

عنصر_مجموعة ::= تعبير : تعبير .. تعبير

عنوان ::= حرف_أو_رقم { حرف_أو_رقم }

عنوان_حالة ::= ثابت

علامة_حقل ::= لا شيء : اسم_تعريف_حقل

قائمة_اسم_تعريف ::= اسم_تعريف { ، اسم_تعريف }

قائمة_اسم_تعريف_ملف ::= لا شيء : (اسم_تعريف_ملف { ،
اسم_تعريف_ملف })

قائمة_حالة ::= عنصر_قائمة_حالة { ، عنصر قائمة حالة }

قائمة_حيث ::= قيمة_مبدئية الى قيمة_نهائية : قيمة_مبدئية تنازلي قيمة_نهائية

قائمة_حقل ::= منطقة_محددة : منطقة_محددة؛ منطقة_متنوع: منطقة_متنوع

قائمة_عنصر_آني ::= صحيح_بدون_اشارة : اسم_تعريف_ثابت :
اسم_تعريف_متغير : اسناد_عداد_موقع

قائمة_عنوان_حالة ::= عنوان_حالة { ، عنوان_حالة }

قائمة_متغير_سجل ::= متغير_سجل { ، متغير_سجل }

قسم_تعريف ::= منطقة_تعريف_عنوان : منطقة_تعريف_ثابت :
منطقة_تعريف_صنف : منطقة_تعريف_متغير :
منطقة_تعريف_اجراءات_ودوال

قسم_سجل ::= لا شيء : اسم_تعريف_حقل { ، اسم_تعريف_حقل }: صنف

قسم_معطيات_اساسية ::= مجموعة_معطيات : **متغير** مجموعة_معطيات

قيمة_مبدئية ::= تعبير

قيمة_نهائية ::= تعبير

كتلة ::= منطقة_تعريف منطقة_جمل

لا شيء ::=

متغير ::= متغير_كلي : متغير_عضو : متغير_مسند

متغير_تحكم ::= اسم_تعريف_متغير

متغير_سجل ::= متغير

متغير_عضو ::= متغير_مفهرس : مخصص_حقل

متغير_كلي ::= اسم_تعريف_متغير : اسم_تعريف_ثابت_مصنف

متغير_مفهرس ::= متغير_منظومة [تعبير { تعبير }]

متغير_مسند ::= متغير_مؤشر ^

متغير_مؤشر ::= متغير

متغير_منظومة ::= متغير

مجموعة ::= [{ عنصر_مجموعة }]

مجموعة_ثابتة ::= [{ عنصر_مجموعة_ثابتة }]

مجموعة_معطيات ::= قائمة_اسم_تعريف : صنف_اسم_تعريف

متنوع ::= لا شيء : قائمة_عنوان_حالة : (قائمة_حقل)

مخصص_حقل ::= متغير_سجل. اسم_تعريف_حقل

مخصص_دالة ::= اسم_تعريف_الدالة : اسم_تعريف_دالة (معطيات_فعلية { ، معطيات_فعلية })

مقدمة_اجراء ::= **اجراء** اسم_تعريف؛ : **اجراء** اسم_تعريف (قسم_معطيات_اساسية { ، قسم_معطيات_اساسية })؛

مقدمة_برنامج ::= لا شيء : **برنامج** اسم_تعريف_برنامج قائمة_اسم_تعريف_ملف

مقدمة_دالة ::= **دالة** اسم_تعريف : صنف_نتيجة؛ : **دالة** اسم_تعريف (قسم_معطيات_اساسية { ، قسم_معطيات_اساسية }) : صنف_نتيجة؛

مصطلح ::= معامل_متمم { عامل ضرب معامل متمم }

معامل ::= متغير : ثابت_بدون_اشارة : (تعبير) : مخصص_دالة : مجموعة

معامل_باشارة ::= معامل : اشارة معامل

معامل_قياس ::= تتابع_رقم : اشارة تتابع_رقم

معامل_متممُ ::= معامل_باشارة : **عكس** معامل_باشارة

معطيات_فعلية ::= تعبير : متغير

منطقة_تعريف_ثابت ::= **ثابت** تعريف_ثابت { ؛ تعريف_ثابت }؛

منطقة_تعريف ::= { قسم_تعريف }

منطقة_محددة ::= قسم_سجل { ؛ قسم_سجل }

منطقة_تعريف_عنوان ::= **عنوان** عنوان { ، عنوان }؛

منظومة_ثابتة ::= (ثابت_مهيكل { ثابت_مهيكل})

منطقة_تعريف_اجراء_ودالة ::= { تعريف_اجراء_أو_دالة }

منطقة_جملة ::= جملة_مركبة

منطقة_تعريف_صنف ::= **صنف** تعريف_صنف { ؛تعريف_صنف }؛

منطقة_تعريف_متغير ::= **متغير** تعريف_متغير { ؛ تعريف_متغير }؛

منطقة_متنوع ::= **حالة** علامة_حقل اسم_تعريف_صنف **من** متنوع {؛ متنوع}

الفهـــرس

ـ أ ـ

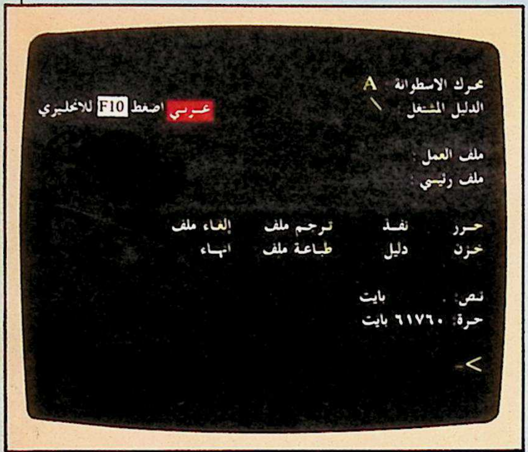
— ب —

ـ ن ـ

— و —

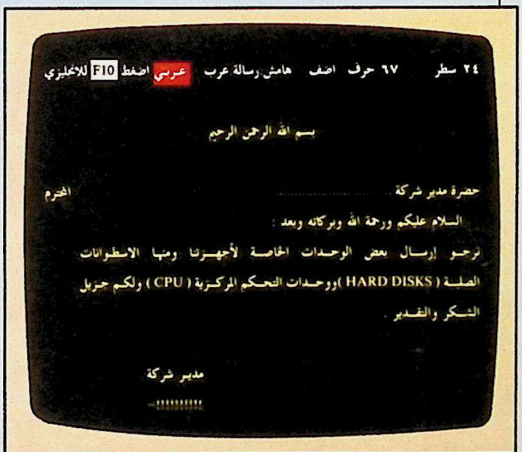
نبراس

- لغة برمجة عربية ١٠٠ ٪ وسهلة الاستخدام .
- لغة هيكلية .
- تعمل على أجهزة آي . بي . إم والمتوافقة معها .
- عمل برامج تنفيذية يتم تنفيذها مباشرة من نظام التشغيل.
- إدخال وإخراج البيانات باللغة العربية والانجليزية .
- معالجة الملفات البيانية :
- عمل جميع أشكال الرسم البيانية والهندسية .



القائمة الرئيسية :

تسهل للمبرمج استعمال اللغة حيث تحتوي على أوامر رئيسية يتم تنفيذها بالضغط على مفتاح واحد فقط .



المحرر :

معالج كلمات داخلي عربي / انجليزي مطور بدون الاستعانة بأي برنامج عربي .



المترجم :

تنفيذ سريع للبرامج وتحديد الأخطاء الناتجة في البرنامج وطريقة تصحيحها .

مطابع الشريف